U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards

NBS Special Publication 724

Real-Time Optimization in Automated Manufacturing Facilities

Richard H.F. Jackson and Albert W.T. Jones, Editors

National Bureau of Standards

The National Bureau of Standards[1] was established by an act of Congress on March 3, 1901. The Bureau's overall goal is to strengthen and advance the nation's science and technology and facilitate their effective application for public benefit. To this end, the Bureau conducts research and provides: (1) a basis for the nation's physical measurement system, (2) scientific and technological services for industry and government, (3) a technical basis for equity in trade, and (4) technical services to promote public safety. The Bureau's technical work is performed by the National Measurement Laboratory, the National Engineering Laboratory, the Institute for Computer Sciences and Technology, and the Institute for Materials Science and Engineering.

The National Measurement Laboratory

Provides the national system of physical and chemical measurement; coordinates the system with measurement systems of other nations and furnishes essential services leading to accurate and uniform physical and chemical measurement throughout the Nation's scientific community, industry, and commerce; provides advisory and research services to other Government agencies; conducts physical and chemical research; develops, produces, and distributes Standard Reference Materials; and provides calibration services. The Laboratory consists of the following centers:

- Basic Standards[2]
- Radiation Research
- Chemical Physics
- Analytical Chemistry

The National Engineering Laboratory

Provides technology and technical services to the public and private sectors to address national needs and to solve national problems; conducts research in engineering and applied science in support of these efforts; builds and maintains competence in the necessary disciplines required to carry out this research and technical service; develops engineering data and measurement capabilities; provides engineering measurement traceability services; develops test methods and proposes engineering standards and code changes; develops and proposes new engineering practices; and develops and improves mechanisms to transfer results of its research to the ultimate user. The Laboratory consists of the following centers:

- Applied Mathematics
- Electronics and Electrical Engineering[2]
- Manufacturing Engineering
- Building Technology
- Fire Research
- Chemical Engineering[2]

The Institute for Computer Sciences and Technology

Conducts research and provides scientific and technical services to aid Federal agencies in the selection, acquisition, application, and use of computer technology to improve effectiveness and economy in Government operations in accordance with Public Law 89-306 (40 U.S.C. 759), relevant Executive Orders, and other directives; carries out this mission by managing the Federal Information Processing Standards Program, developing Federal ADP standards guidelines, and managing Federal participation in ADP voluntary standardization activities; provides scientific and technological advisory services and assistance to Federal agencies; and provides the technical foundation for computer-related policies of the Federal Government. The Institute consists of the following centers:

- Programming Science and Technology
- Computer Systems Engineering

The Institute for Materials Science and Engineering

Conducts research and provides measurements, data, standards, reference materials, quantitative understanding and other technical information fundamental to the processing, structure, properties and performance of materials; addresses the scientific basis for new advanced materials technologies; plans research around cross-country scientific themes such as nondestructive evaluation and phase diagram development; oversees Bureau-wide technical programs in nuclear reactor radiation research and nondestructive evaluation; and broadly disseminates generic technical information resulting from its programs. The Institute consists of the following Divisions:

- Ceramics
- Fracture and Deformation [3]
- Polymers
- Metallurgy
- Reactor Radiation

[1] Headquarters and Laboratories at Gaithersburg, MD, unless otherwise noted; mailing address Gaithersburg, MD 20899.
[2] Some divisions within the center are located at Boulder, CO 80303.
[3] Located at Boulder, CO, with some elements at Gaithersburg, MD.

NBS Special Publication 724

Real-Time Optimization in Automated Manufacturing Facilities

Proceedings of a Symposium Held at the National Bureau of Standards, Gaithersburg, MD, January 21-22, 1986

Edited by:

Richard H. F. Jackson
Center for Applied Mathematics
National Bureau of Standards
Gaithersburg, MD 20899

and

Albert W. T. Jones
Center for Manufacturing Engineering
National Bureau of Standards
Gaithersburg, MD 20899

Sponsored by:

The Center for Manufacturing Engineering and
The Center for Applied Mathematics of
The National Bureau of Standards
Gaithersburg, MD 20899

and

U.S. Department of the Navy
Manufacturing Technology Program
Washington, DC 20360

Issued September 1986

U.S. Department of Commerce
Malcolm Baldrige, Secretary
National Bureau of Standards
Ernest Ambler, Director

Library of Congress
Catalog Card Number: 86-600580
National Bureau of Standards
Special Publication 724
Natl. Bur. Stand. (U.S.),
Spec. Publ. 724
481 pages (Sept. 1986)
CODEN: XNBSAV

U.S. Government Printing Office
Washington: 1986

For sale by the Superintendent
of Documents,
U.S. Government Printing Office,
Washington DC 20402

FOREWORD

The National Bureau of Standards (NBS) has established an experimental testbed, the Automated Manufacturing Research Facility (AMRF) at the NBS site in Gaithersburg, MD. The purpose of the AMRF is to support the efforts of American industry to compete in the "Modern Industrial Revolution." It will provide both a facility which can be used to develop and test automated manufacturing standards, and mechanisms for transferring that technology to the American marketplace. The AMRF will serve as an engineering testbed that researchers from NBS, universities, industry and other government agencies can use to study questions of standardization, measurement, and quality control in the automated factory environment. In fact, there is already a high level of participation by universities and industry.

By the end of 1986, the AMRF will contain several robot-tended work centers, including machining, cleaning and deburring, automated inspection, and material transfer. These individual work centers will be linked by a modular, hierarchically structured planning and control system and a sophisticated data management and communication system. The AMRF will thus demonstrate that it is possible to transform a collection of manufacturing and computer equipment purchased from several different vendors into a small, fully integrated, flexible manufacturing system.

As indicated above, the AMRF has been established to support the efforts of American manufacturers to increase productivity. One measure of productivity is the actual throughput of the manufacturing plant, which is directly related to the planning, scheduling, and routing strategies employed at the plant. Although much has been written on theoretical solutions to these problems, results of practical value have appeared less often. Realizing this, researchers at the AMRF have embarked on an effort to develop better solution techniques and to test these techniques thoroughly in the real-life environment provided by the AMRF.

An important first step in this effort was to gain a full understanding of the state-of-the-art. This was the primary motivation for holding the Symposium on Real-Time Optimization in Automated Manufacturing Facilities, held at the National Bureau of Standards, January 21-22, 1986. This symposium brought together those who design and test solution procedures with those who must implement and use them in a real manufacturing environment. In addition, it provided an opportunity for participants to become acquainted with the first national testbed for analyzing the efficiency of these procedures, the National Bureau of Standards' Automated Manufacturing Research Facility.

These proceedings contain most of the papers presented at that symposium, and represent, we believe, an important and valuable contribution to the literature on planning, scheduling and routing problems in automated manufacturing.

Richard H. F. Jackson
Albert W. T. Jones

Gaithersburg, MD
September 1986

ABSTRACT

The Symposium on Real-Time Optimization in Automated Manufacturing Facilities was held at the National Bureau of Standards, Gaithersburg, Maryland, January 21-22, 1986. It was jointly sponsored by the Center for Manufacturing Engineering (with funds obtained from the Navy Manufacturing Technology Program) and the Center for Applied Mathematics. It was designed to bring together those who design and test optimization procedures for solving planning, scheduling, and routing problems with those who must use these procedures in a real-time manufacturing environment. Included in the proceedings are discussions of the following topics: an approach to hierarchical production planning and scheduling; a hierarchy of intelligent scheduling and control for automated manufacturing systems; the integration of planning scheduling, and control for automated manufacturing; intelligent manufacturing planning systems; a decision making framework for manufacturing systems; PATRIARCH -- hierarchical production scheduling; low-level interactive scheduling; the general employee scheduling problem: an effective large scale solution approach; ISIS project in review; dynamic control in automated manufacturing: a knowledge integrated approach; a management control approach to the manufacturing, planning, and scheduling problem; hierarchies of sub-periods in constraint-directed scheduling; a two-level planning and scheduling approach for computer integrated manufacturing; match-up real-time scheduling; a maximal covering model for loading flexible manufacturing systems; interstage transportation planning in the flow-shop environment; minimal technology routing and scheduling systems based on space filling curves; a multi-pass expert control system (MPECS) for flexible manufacturing systems; requirements for automatic control of aerospace manufacturing processes; real-time scheduling of an automated manufacturing center; the interaction between design and scheduling in repetitive manufacturing environments; an adaptable scheduling algorithm for flexible flow lines (abstract); determining aggregated FMS production ratios and minimum inventory requirements; a knowledge based system for dynamic manufacturing replanning; dispatching -- the critical automation link; scheduling jobs in flexible manufacturing systems; design requirements for a real-time production scheduling decision aid; an optimized Kanban system for a custom door manufacturer; a linear programming approach to numerically controlled face milling; and, system buffers required when FMS are used in fabrication and assembly operations.

KEYWORDS:

automated manufacturing, flexible manufacturing, hierarchical control, planning problems, routing problems, real-time optimization, scheduling problems.

TABLE OF CONTENTS

AN APPROACH TO HIERARCHICAL PRODUCTION PLANNING AND SCHEDULING

by

Stanley B. Gershwin

Laboratory for Information and Decision Systems
35-433
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

Introduction

The fundamental issue in scheduling is that the most obvious formulations (combinatorial, integer programming) are not computationally feasible. That is, no algorithm will ever be implemented that will provide a truly optimal solution to a non-trivial problem when it is modeled by such techniques. All scheduling methods must deal with this issue in some way.

Our approach is to decompose the problem into a set of related problems. The decomposition will proceed in two dimensions: by time scale and by floor space.

In the floor space decomposition, the system is divided into a set of cells or workstations. Statistical models of arrivals and demands are assumed and then important quantities--such as a feedback law for scheduling--are calculated within the cell. The resulting departures from that cell become arrivals at other cells; and the loading of parts into that cell translates into demands for other cells. When this is all made explicit, it leads to a complex set of equations relating the behavior of each cell to that of each other.

The time scale decomposition is a hierarchy in which higher levels deal with longer range and more aggregated issues, and lower levels deal with short term and highly specific issues.

Hierarchy

There are many time scales over which planning and scheduling decisions must be made. While these decisions are made separately, they are related. In particular, each long term decision presents an assignment to the next shorter term deci-

sion-maker. The decision must be made in a way that takes the resources--i.e., the capacity--explicitly into account.

Figure 1 shows a schematic of an information and material flow system in a factory. The information flow system is hierarchically organized, consisting of modules as shown in Figure 2. Each module has requirements specified at the higher levels. It solves an optimization problem to calculate requirements to be issued to the lower levels. The optimization problem is one in which the objective is to meet specified requirements as closely as possible, subject to current resources and capacity constraints. Uncertainties in supply and demand are reported by neighboring modules.

In this scheme, higher levels deal with longer time scales, more floor space, and less detail than lower levels. The most popular time-scale decomposition approach in control theory is the singular perturbation approach (Kokotovic and O'Malley, 1976; Kokotovic, 1984; Coderch et al., 1983; Lou et al., 1984; Chow, 1978; Haddad, 1976). This technique reduces the model order by separating the time scales, that is, by considering slow and fast phenomena separately. There are two purposes in doing this. First, a large problem can be replaced by several smaller problems so that the computational burden can be reduced. Second, this kind of control is closer to the real situation where managers at different levels are only concerned about their own specific time scales.

These are useful concepts upon which to base a hierarchical scheduling scheme. However, it is not likely that existing mathematical techniques can be used for this without substantial development.

Scheduling is simplified by dealing with separate issues at different time scales. In particular, capacity depends on the time scale over which it is considered. For it to have any meaning, capacity must remain constant over the time required for the dispatch of several parts. We have found (Akella, Choong, and Gershwin, 1984) that we could very effectively schedule a detailed model of a flexible manufacturing system by

1. choosing a target production rate at a higher level of a hierarchical scheduling algorithm, and then

2. dispatching parts in a way that maintains the actual production rate close to the target.

Figure 2 represents a scheduling hierarchy. It is based on a specific set of assumptions about time scales. Other hierarchies are appropriate when other sets of assumptions are valid. Note that each level may itself consist of several levels. In addition, this is only part of the hierarchy. At still higher levels are issues related to capital expansion and the national economy. At lower levels are the detailed instructions that guide a robot arm, and that pass information among machines

connected in a communication network. (See, for example, Barbera, Fitzgerald, and Albus, 1982.)

It is assumed that parts can be grouped into families such that within the families, setup times are negligible. The setup times required to change from one family to another are assumed to be large. The mean times between failures and the mean times to repair machines are assumed to be large compared with the times to do operations on parts, and the setup times in changing from one family to another are assumed to be large compared with the repair and failure times. Finally, the overall planning horizon is large compared with the inter-family setup times.

In Figure 2, the top level assures that the total volume of orders is within system capacity and the second level coordinates cells. In the second level, the system is divided into a set of cells or workstations. The departures from a cell are arrivals at other cells; and the loading of parts into that cell translates into demands for other cells. A feedback law is calculated to determine long term target production rates as a function of demand uncertainties. The feedback law for the factory as a whole may be similar.

The third level calculates production time (hours per day that the facility will be run), setup frequencies, fraction of time each machine will spend in each of its configurations, and production rate of each part type in each configuration. Setups are an important issue because the time spent setting up is time taken away from producing and thus setups reduce capacity. However, infrequent setups lead to large delays and excess WIP (Work-In-Process). This level is discussed below.

At the fourth level, the actual times at which to change setups are calculated. These times are chosen in a way that is consistent with the setup frequencies calculated at the third level, but that is also responsive to events observed in real time, such as a run of bad luck in which a machine is down more than is typical.

The fifth level calculates the best mix of production rates as a function of past production (i.e., is the system ahead or behind nominal production?) and current machine states (operational, failed, being maintained, etc.) for the current setup.

The sixth level dispatches parts into the system in a way that is consistent with the production rate calculation of the fifth level.

Capacity

Demands must be within capacity or excessive queuing will occur, leading to excessive costs, delays, and poor morale. However, capacity is difficult to define and evaluate. One reason is that the definition of capacity depends on the time scale. For example, short-time-scale (fifth level) capacity is a func-

tion of the set of machines operational at any instant. Long-time-scale capacity is an average of short-time-scale capacity.

Other reasons for the difficulty in specifying a system's capacity include the effects of set-up times, the effects of randomness and buffers, and the uncertainty in system parameters.

Hierarchically Structured Production Planning

Hax and Meal (1975) and Bitran and Hax (1977) proposed a hierarchical production planning approach which aggregates production units into items, families, and types. It starts from aggregate planning using linear programming. The aggregate plan is then broken down (disaggregated) into schedules for families and items. The underlying idea of their approach is to make decisions sequentially starting from the highest level. The decision at each level then becomes a constraint for the next lower level. In this way, not only is the computation burden reduced but also the decision made in each level is consistent with its own management responsibilities.

Maxwell et al. (1983) present a three-level system. The first level determines the long-term plan. The middle level deals with uncertainties setting safety stock in appropriate places. The lowest level takes care of detailed resource allocation.

There have been many other hierarchical structures proposed as well, including several at the NBS Symposium. However, much of the previous work is not dynamic. It does not respond to random events such as new demands, demand cancellations, raw material shortages, machine failures, and so on, in a way that is based on explicit stochastic, dynamic models.

An exception is the work of Kimemia (1982), Kimemia and Gershwin (1983), and Gershwin, Akella, and Choong (1985). A hierarchical scheduling policy was developed and applied to the scheduling of a simulation of an IBM printed circuit board manufacturing system. The main feature of this approach was to use the available information and system flexibility efficiently to anticipate and to react to disruptive events such as machine failures.

The time scale assumptions that underlie this algorithm are the following: setup times are negligible compared with operation times; operation times are short compared with failure and repair times; and failure and repair times are short compared with the planning horizon. The relaxation of the first assumption is discussed below. That algorithm is also based on the assumption that demand is deterministic and raw material is always present, and that assumption is also relaxed below.

The hierarchical structure of the Kimemia-Gershwin scheduling policy is shown in Figure 3 This algorithm may be viewed as

the two lower levels in Figure 2. The middle level of Figure 3 determines the short-term production rates, taking the capacity constraints of the system into account. Based on these rates the lower level determines the actual times at which parts are loaded into the system. The middle level uses immediate machine status and surplus/backlog information for its computations. It also needs certain longer term information, supplied by the higher level, such as failure and repair rates, and part data such as operation times and demand.

Simulation results (Akella, Choong, and Gershwin, 1984) have shown that the Figure 3 hierarchical structure is effective in scheduling an FMS. It can achieve high output with low WIP and cope with changes and disturbances.

The major insight gained from this work was the simplification of the scheduling task by separating the capacity issues from the dispatch issues. Scheduling is difficult because every time a decision is made about sending a part into a system, there are two kinds of questions to consider: how does this decision affect the production of the part being dispatched?; and how does this decision affect the rest of the system?

Kimemia and Gershwin solved this problem by treating the latter question at a higher level in the hierarchy than the former. The effect on the rest of the system is treated by considering the system's capacity as a function of the current set of operational machines. A time-varying production rate target for each part type is calculated which is within the current capacity. The lower level's responsibility is then merely to dispatch parts according to the target production rate. Because the total production rate is guaranteed to be within the capacity, the problem is greatly simplified.

First and Second Level Scheduling-- Supply and Demand Uncertainties

Aggregate planning determines the production quantities and the material orders over a time horizon of several months. The objectives of this level are to minimize the inventory cost both in raw material and in products while keeping the production close to some specified rate.

Using linear programming for aggregate planning has a long history (McClain and Thomas, 1985). Demand is treated as deterministic (using forecast instead of real demand). In spite of the approximation of many features of the problem, the computational load is heavy. The effect of the planning horizon has been studied by McClain and Thomas (1977). Other extensions, e.g. productivity of workers changing with time (Ebert, 1976), and non-continuous production rate change (Hillier and Lieberman, 1980) have also been considered. In all these papers, the supply and demand were considered deterministic.

In earlier aggregate planning work by Holt et al. (1955,

1956), closed form results were developed based on the concept of feedback control of a linear system with quadratic cost. Recently, there has been more effort devoted to using traditional control theory to production planning (Sethi and Thompson, 1981, Clifford, 1985).

Both supply and demand are highly random. New orders may be added and old ones may be canceled. Raw materials may come late or early. The approach proposed here is to make full use of the probabilistic structure of the demand and supply. Combined with the results of control theory we believe a robust scheduling technique can be constructed.

Preliminary results indicate that random demand can be incorporated into the system dynamics. Standard results of stochastic control theory show that, under certain conditions, the optimal production rate p(t) can be expressed as

$$p(t) = K(t)x(t)$$

Here, p(t) satisfies capacity constraints and is chosen to minimize the inventories in the system; and x(t) is a vector related to raw material inventory, product inventory, and demand. The feedback gain matrix K(t) is determined by system parameters.

Third Level Scheduling--Set-Ups

Some time is always required to change the configuration of the system from making one part type (or family of part types) to making another. This is an issue of practical importance which arises in metal cutting systems, in which tool magazines can hold only a limited number of tools; printed circuit card assembly systems; and VLSI fabrication, in which furnaces must be cleared of each kind of impurity before the next kind of impurity can be used.

It is not desirable to change the configuration too often because that reduces the amount of time available for productive work and thus the capacity of the manufacturing system. On the other hand, it is not desirable to change it too infrequently, because that will tend to increase inventories and delays before deliveries.

Instead of calculating precise times at which to change setups, we calculate guidelines to be carried out by a lower level algorithm. That is, we specify frequencies of changeover, and it is the responsibility of the lower level algorithm to choose the actual times at which to perform the changeovers. The times may be determined by the clock, or by events such as the production of a given quantity of material. As long as the triggering events lead to frequencies that are consistent with what is calculated at the higher level, satisfactory behavior will be observed. This approach is described in detail in Gershwin (1986).

This approach to scheduling differs from conventional mixed integer programming representations (Graves, 1981). In such formulations, there are a large number of binary or integer variables that represent whether or not a given part is to be produced at a given machine at a given time (for example, Karmarkar and Schrage, 1985). These approaches model the system in detail, but their large computational requirements make them difficult to solve and interpret, and adding stochastic phenomena does not make them any easier. Maxwell and Muckstadt (1985) simplify the problem by dealing only with reorder intervals (ie setup frequencies) and ignoring capacity questions. Kusiak, Vanelli, and Kumar (1985) treat only the grouping problem.

The goal of the analysis is to calculate the long-term average frequencies of set-ups and the average fraction of time that the system is in each configuration (i.e., the fraction of time it is set up for each family of part types). Further work is required to translate these quantities into times at which to perform set-ups.

The approach is to view the configurations of the FMS as similar to states in a Markov process. The system spends a random amount of time in each configuration, and we seek to determine the rates at which the system moves from each configuration to each other. (The inverses of these rates are the average lengths of intervals during which the system is in each configuration.) When the rates are known, the fraction of time in each configuration is known.

There are two key equations in Markov chain analysis. The first is the Kolmogorov equation, which relates the fraction of time the system spends in each state to the rates of flow from state to state. The second is the normalization equation, which requires that the sum of the fractions of time in each state be 1.

In using this theory to characterize setup times in a manufacturing system, the normalization equation must be modified. The sum of the fractions of time that a manufacturing system spends in each configuration is less than 1 because it may spend a substantial amount of time changing configurations: setting up. The normalization equation must therefore include a term which accounts for the setup time.

It is this term that accounts for the reduction in system capacity if setups are too frequent. When setup frequencies go up, that term increases. The sum of the fractions of time that the system spends working in each configuration therefore goes down. When setups are frequent, capacity is small. When setups are infrequent, batch sizes and therefore inventory are large.

This leads to an optimization problem which balances the needs to keep capacity high (to maximize the probability that the required amounts of material are actually produced) and inventory low.

Fourth Level Scheduling--Setup Instants

At the third level, the frequencies of changing configurations are determined, as well as the fractions of time the system appears in each configuration and the production rates for each member of the family associated with each configuration. This does not tell the whole story, however; a rule is needed to determine exactly when to change configurations. This rule must be such that the actual changes take place in a way that is consistent with the quantities at the third level.

For example, one approach would be to perform setups according to the clock, with setup times at fixed instants chosen to be consistent with the frequencies calculated at the higher level. This may lead to the correct average production rates in the long run. However, because of repairs and failures, the amount of material produced during a period while the system is in some configuration may deviate by a great deal from the amount required at the third level. An alternative would be to perform setups at times dictated by the cumulative amounts of material produced, but this must be done with care since each family may consist of many parts, some of which may be in surplus at any given time, and some in backlog.

Fifth and Sixth Levels--Response to Machine Failures

At the fifth level of the hierarchy is the response of the system to failures of machines. This is included in the top two levels of Figure 3. At the top level of this sub-hierarchy, a dynamic programming calculation is performed to find the feedback law that is implemented at the middle level. The feedback law is the solution to a small linear programming problem. Its output is the production rate of each part type (in the family currently configured) as a function of the current set of operational machines, and the cumulative production of all the part types in the family.

Although the fifth level calculates production rates, it still does not specify exactly when a part must be dispatched into the system. The sixth level (depicted as the bottom level of Figure 3) has this responsibility. This is analogous to the need for the fourth level to select setup instants after the third level calculates setup frequencies. A simple, but evidently very effective policy is described in Gershwin, Akella, and Choong (1985).

Conclusion

As manufacturing systems become more and more automated, the rules for determining when important actions (such as loading parts, beginning operations, and performing setups) must be made more and more explicit. The only feasible way to calculate an effective set of rules is to decompose the manufacturing system. A hierarchical decomposition is described here which is based on

calculating the system's capacity at various levels, and developing explicit policies to respond to uncontrollable changes in the system such as machine failures.

REFERENCES

R. Akella, Y. F. Choong, and S. B. Gershwin (1984), "Performance of Hierarchical Production Scheduling Policy," IEEE Transactions on Components, Hybrids, and Manufacturing Technology, Vol. CHMT-7, No. 3, September, 1984.

G. Bitran and A. Hax (1977), "On the Design of Hierarchical Production Planning Systems, Decision Sciences, Volume 8, Number 1, pp. 28-55.

J. H. Chow (1978), "A Newton-Lyapunov Design for a Class of Nonlinear Regulator Problems," Proc. 16th Allerton Conf. on Communication, Control and Computing.

M. Clifford (1985), "Control Theoretic Approach to Planning Models for Hybrid Manufacturing Systems," Technical Report for Center for Manufacturing Productivity, Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, New York.

M. Coderch, A. Willsky, S. Sastry, and D. Castanon (1983), "Hierarchical Aggregation of Linear Systems with Multiple Time Scales," IEEE Trans. Automatic Control, Vol. AC-28, No. 11, 1983.

R. J. Ebert (1976), "Aggregate Planning with Learning Curve Productivity," Management Science, Vol. 23 (1976), pp 171-182.

S. B. Gershwin (1986), "Stochastic Scheduling and Setups in Flexible Manufacturing Systems," MIT Laboratory for Information and Decision Systems Report, April, 1986.

S. B. Gershwin, R. Akella, and Y. F. Choong (1985), "Short-Term Production Scheduling of an Automated Manufacturing Facility," IBM Journal of Research and Development, Vol. 29, No. 4, pp 392 400, July, 1985.

S. C. Graves (1981), "A Review of Production Scheduling," Operations Research, Vol. 29, No. 4, pp 646-675, July-August, 1981.

A. H. Haddad (1976), "On Singular Perturbations in Stochastic Dynamic Systems," 10th Asilomar Conf. on Circuits, Systems, and Computers, Nov. 1976.

A. Hax and H. Meal (1975), Hierarchical Integration of Production Planning and Scheduling, in TIMS Studies in the Management Sciences, Vol. 1: Logistics, M. Geisler, ed. New York: North-Holland/American Elsevier, 1975.

F. S. Hillier and G. J. Lieberman (1980), Introduction to Operations Research, 3rd ed. San Francisco: Holden-Day, 1980.

C. Holt, F. Modigliani, and H. Simon (1955), "A Linear Decision Rule For Production and Employment Scheduling," Management Science, Vol. 2, No. 1, Oct. 1955, pp 1-30.

C. Holt, F. Modigliani, and J. Muth, (1956), "Derivation of a Linear Decision Rule for Production and Employment Scheduling," Management Science, Vol. 2, No. 2, Jan. 1956, pp 159-177.

U. S. Karmarkar and L. Schrage (1985), The Deterministic Dynamic Product Cycling Problem," Operations Research, Volume 33, No. 2, March-April 1985, pp 326-345.

J. G. Kimemia (1982), "Hierarchical Control of Production in Flexible Manufacturing Systems," MIT Laboratory for Information and Decision Systems Report LIDS-TH-1215.

J. Kimemia and S. B. Gershwin (1983), "An Algorithm for the Computer Control of a Flexible Manufacturing System," IIE Transactions Vol. 15, No. 4, pp 353-362 December, 1983.

A. Kusiak, A. Vanelli, and K. R. Kumar (1985), "Grouping Problem in Scheduling Flexible Manufacturing Systems," Robotica, Volume 3, pp 245-252.

P. V. Kokotovic (1979), "An Introduction to Singular Perturbations," Systems Engineering for Power, Organizational Forms for Large Systems, Vol. 2, Oct., 1979.

P. V. Kokotovic, R. E. O'Malley, Jr., and P. Sannuti (1976), "Singular Perturbation and Order Reduction in Control Theory--An Overview," Automatica, Vol. 12, pp. 123-132, 1976.

S. X. C. Lou, G. Verghese, A. S. Willsky and M. Vidyasagar (1984), "An Algebraic Approach to Analysis and Control of Time-Scales," American Control Conf., San Diego, 1984.

W. Maxwell and J. Muckstadt (1985), "Establishing Consistent and Realistic Reorder Intervals in Production-Distribution Systems," Operations Research, Volume 33, No. 6, November-December 1985, pp 1316-1341.

W. Maxwell, J. Muckstadt,J. Thomas, and J. VanderEcken (1983), "A Modeling Framework for Planning and Control of Production in Discrete Parts Manufacturing and Assembly Systems," Interfaces, Volume 14, Number 6, December, 1983.

J. O. McClain and L. J. Thomas (1977), "Horizon Effects in Aggregate Production Planning with Seasonal Demand, Management Science, Vol. 23, No. 7, pp 728-736, March, 1977.

J. O. McClain and L. J. Thomas, (1985), Operations Management,

Production of Goods and Services, Prentice-Hall, 1985.

S. Sethi and G. Tompson (1981), "Simple Models in Stochastic Production Planning," in Applied Stochastic Control in Econometrics and Management Science, edited by A. Bensoussan et al, North Holland Publishing, pp 295-304.

ACKNOWLEDGMENTS

Sponsored by the Defense Advanced Projects Agency and monitored by the Office of Naval Research under Contract N00014-85K-0213. I am grateful for the suggestions of Dr. Sheldon X.-C. Lou.

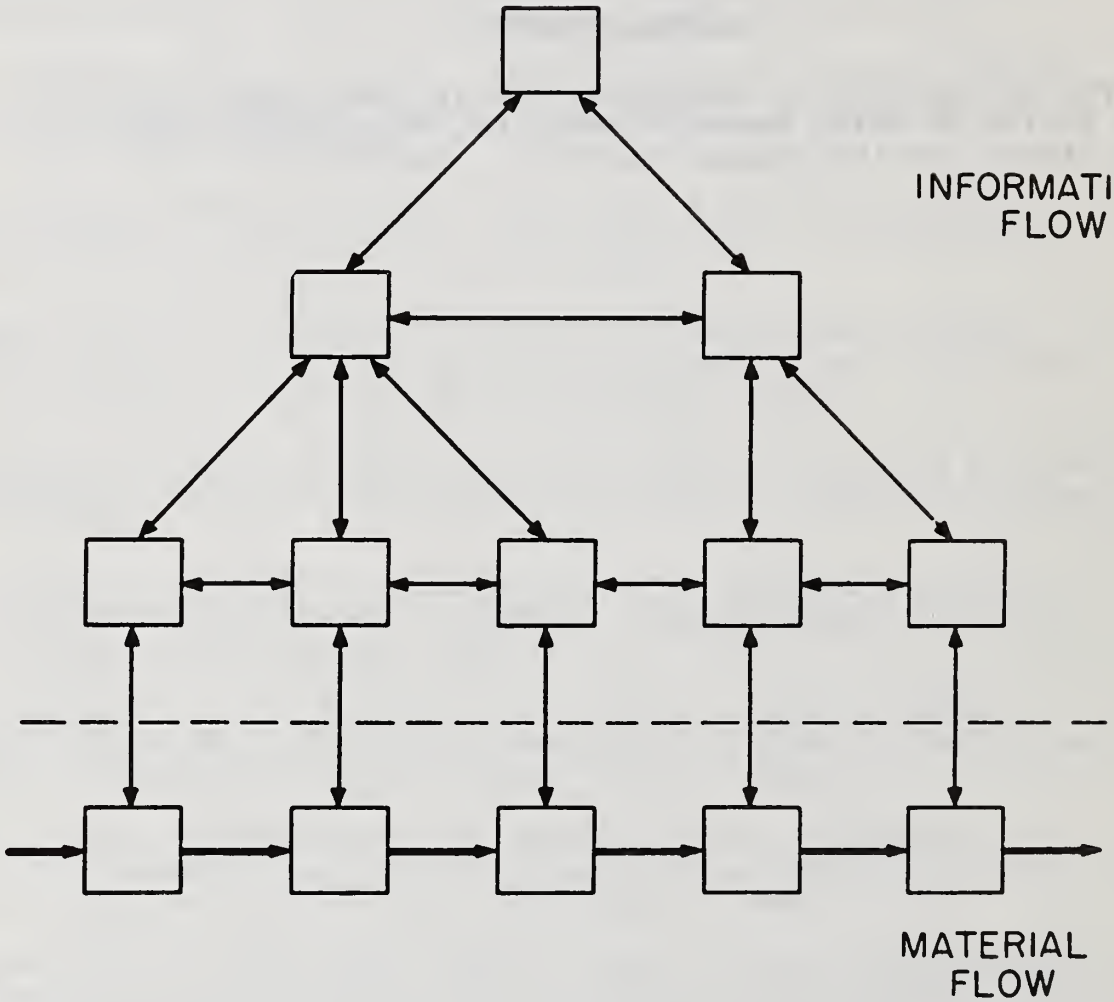


Fig. 1. Hierarchical Structure of Production Planning, Scheduling, and Control System

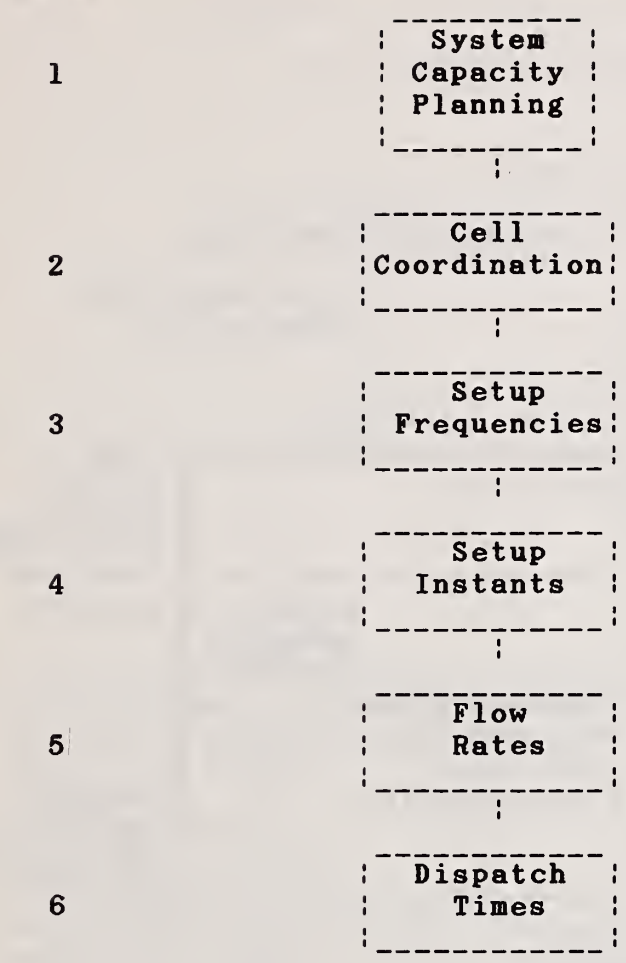


Figure 2. Scheduling Hierarchy

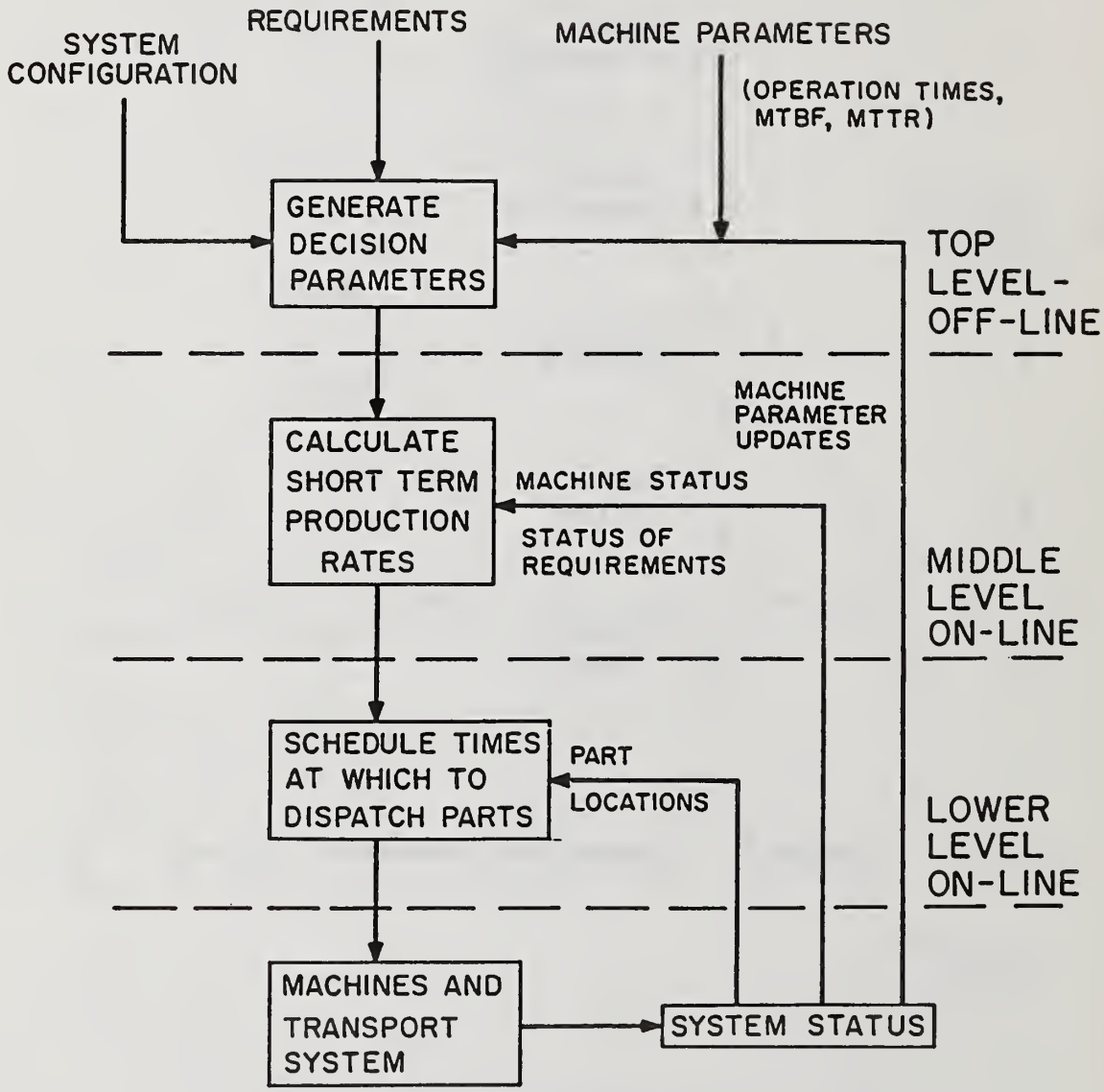


Fig. 3 Hierarchical Algorithm for Short term Production Planning and Scheduling

A HIERARCHY OF INTELLIGENT SCHEDULING AND CONTROL FOR AUTOMATED MANUFACTURING SYSTEMS

Peter J. O'Grady
Department of Industrial Engineering
North Carolina State University
Raleigh, NC 27695-7906

and

Unny Menon
Department of Industrial Engineering
California Polytechnic State University
San Luis Obispo, CA

KEYWORDS

Production control, Artificial Intelligence, hierarchical control, Materials Requirements Planning, production scheduling, automated manufacturing systems, Flexible Manufacturing Systems

1. INTRODUCTION

Automated manufacturing systems consist of a number of NC machines together with the associated material handling devices such as Automated Guided Vehicles, conveyors and robots. These automated manufacturing systems promise lower manufacturing cost for batch manufacture where a variety of products are manufactured in discrete batches. To achieve this lower manufacturing cost a viable and effective scheduling and control system is vital.

The major aim of scheduling and control for automated manufacturing systems is to take the broad system goals and translate them into specific instructions for each item of equipment, responding to problems as and when they occur to reduce their effects. The overall problem is tremendously complex with an enormous number of machine/job/resource combinations to be considered. The approach that can be used to address such a problem is to have a number of levels in a hierarchy of scheduling and control.

This paper gives an overview of the hierarchical approach to scheduling and control of automated manufacturing systems. First the factors affecting scheduling and control of automated manufacturing systems are described and their effect on the design of the scheduling and control system described. Second,

the functions necessary to descend the hierarchy from broad system goals to detailed machine instructions are described. Third, various control hierarchy designs are discussed. Finally, two major functions in the hierarchy, Master Production Scheduling II and On-Line Scheduling are detailed.

2. FACTORS AFFECTING SCHEDULING AND CONTROL OF AUTOMATED MANUFACTURING SYSTEMS

An automated manufacturing system differs considerably from its conventional counterpart. These differences are not only in the manufacturing hardware but are also in the software and communication aspects as well as in the provision of sensors and other monitoring devices. This means automated manufacturing systems have the mechanisms to be closely monitored and controlled so that scheduling and control can be done with more certainty about the actual state of the manufacturing system than with conventional manufacturing systems.

Decisions made in such an environment can therefore be better for three reasons. First, a greater amount of data is available since it is generated automatically. Second, the data produced automatically is likely to be more accurate than that produced by human labor although, of course, a system should be protected against totally ridiculous data points being produced by faulty equipment. The third reason why decisions can be better in an automated environment is that the data can reach the decision making areas faster than in conventional systems.

Although decisions can be made under better circumstances than in conventional systems, there are some features of automated manufacturing systems which can result in a more complex problem to be solved and these features are

(i) Manufacturing lead times are shorter

(ii) Engineering details need to be considered

(iii) Greater emphasis is placed on system utilization.

(iv) There is a need to integrate with existing software systems.

(v) Detailed instructions need to be generated.

Short Manufacturing Lead Times.

One of the major justifications for automating many conventional manufacturing systems is that the manufacturing lead times in automated manufacturing systems can be considerably shorter

than those in conventional manufacturing systems. Achieving this has a number of consequences for scheduling and control. First, there may well be a need for a second Master Production Scheduling (MPS II) function (See section 4).

Second, the speed at which jobs move from operation to operation means that a much more comprehensive on-line control facility is required. This facility must be capable of quickly and accurately determining the next moves to be made in the automated manufacturing system. There may therefore be a need for a faster response time from the scheduling and control system in automated manufacturing systems over the more conventional manufacturing systems. However, lead times in conventional manufacturing systems are gradually declining so that the disparity between automated and conventional manufacturing systems is becoming less.

Engineering Details

The low work in progress levels and low manufacturing lead times associated with automated manufacturing systems means that much room for maneuver has been removed. With conventional manufacturing, considering the use of specialized tools in factory scheduling, for example, is not particularly crucial since jobs can usually wait until such tooling is available. However, this waiting is not desirable within an automated manufacturing system and the scheduling and control procedure may have to include a consideration of such aspects as tooling requirements and jig/fixture requirements. Since the same tooling and/or jigs/fixtures may be required by more than one product type then consideration of the use of tooling and jigs/fixtures may have to be made across the whole manufacturing systems. There is, of course, scope for reducing the problem by incorporating procedures that produce more standard designs relying on standard tooling but there are still likely to be a number of automated manufacturing systems that do have this problem [see, for example, Carrie et al. (1983), O'Grady and Menon (1986)].

System Utilization

The aims of most scheduling and control systems are to:

- achieve low through-put times
- have low work in progress levels
- achieve the job due dates
- achieve high system utilization

In most circumstances these aims are conflicting: achieving job due dates, for example, may mean that system utilization, for parts of the system in any case, is low. The large capital cost of most automated manufacturing systems means that high system utilization is usually considered to be of some priority in con-

sidering production runs. It could be argued that the cost of the automated manufacturing system is a "sunk" cost and that consideration of utilization should not be included in the scheduling and control. However, there are two reasons why utilization may be considered. The first is pragmatic: for many industrial concerns, utilization is used as one performance measure and any scheduling and control system would have to recognize this. The second reason is that system utilization is a measure of throughput that can be obtained in more detail than by purely measuring the output. In this manner, the throughput in machine hours can be obtained.

The desire to achieve high system utilization but still to keep low through-put times, low work in progress levels and to achieve job due dates is not an easy task and can require a more sophisticated scheduling and control system than that used in conventional manufacturing.

Need to Integrate

One major consideration in the design and development of a scheduling and control system for an automated manufacturing system must be that it should integrate naturally with the existing software systems within the organization. These existing software systems may form an integrated whole especially where they are from the same vendor. If that is the case, they very probably have readily useable interfaces between the software systems. On the other hand, the existing software systems may form a rather diverse group especially where they have been procurred from different suppliers or written in-house. If this is the case, then suitable interfacing may be more difficult. Whatever the case, whether the systems form an integrated whole or where they are more diverse, then it is unlikely that an industrial concern would be willing to dispose of all the existing software packages. The automated manufacturing systems scheduling and control software should therefore link in readily with existing software packages within the organization such as Material Requirements Planning, Computer Aided Design and Process Planning. This is an important factor in designing the scheduling and control structures for automated manufacturing systems.

Detailed Instructions Need to be Generated

Human workers, especially the more skilled ones, do not need to be given detailed instructions. Often a broad outline "goal" for them to achieve is all that need be given. For example, a skilled lathe turner will often only need to be given the finished dimensions of a part. He will then often select the base bar stock to be used, the tooling necessary and the speeds/feeds to use when machining. Furthermore, he will also sequence operations at the lathe to finish the part. For automated manufacturing systems however, all the detailed in-

structions have to be generated within the scheduling and control/process planning system. Furthermore, the activities have to be coordinated and scheduled in great detail. The provision of detailed instructions mainly occurs at the lower levels of the hierarchies presented in Section 3.

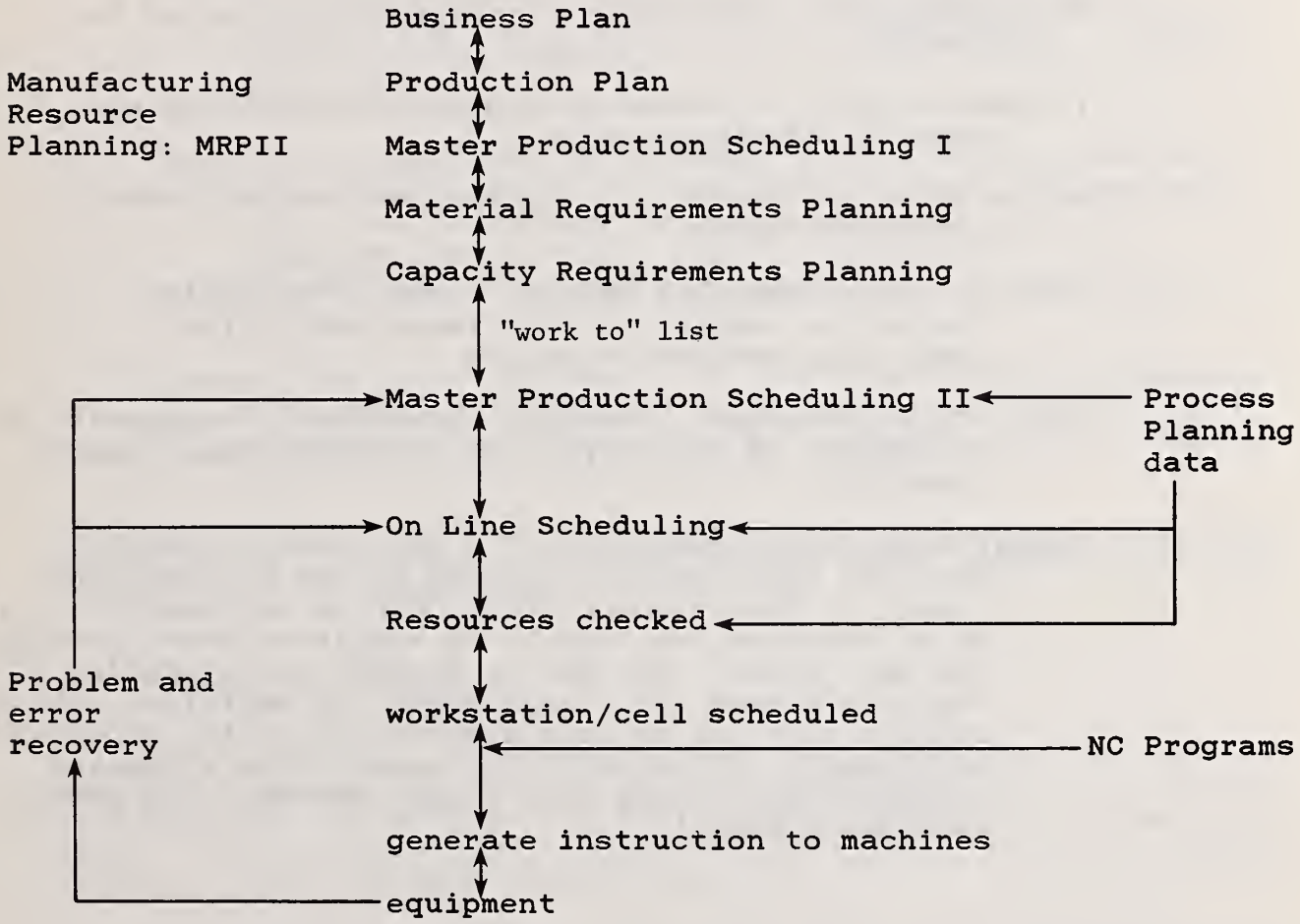


Fig. 1 Example of Functions in Scheduling and Control of Automated Manufacturing Systems

3. FUNCTIONS IN SCHEDULING AND CONTROL

As indicated, the major aim of scheduling and control for automated manufacturing systems is to take the broad system goals and translate them into specific instructions for each item of equipment responding to problems as and when they occur to reduce their effects.

An example of the functions to be carried out as we descend from broad goals to specific instructions is shown in Fig. 1 although not all functions may be present in any particular scheduling and control system.

Business plan - this sets the broad objectives of the company

Production plan - determines aggregate production and expected aggregate sales

Master Production Schedule I - determines the end-product production rates

Material Requirements Planning - breaks the Master Production Schedule I into requirements for components and raw materials

Capacity Requirements Planning - evaluates the capacity requirements of the output for Materials Requirements Planning

Master Production Scheduling II - the output from the Material Requirements Planning system and from the Capacity Requirements Planning is a "work to" list which contains the jobs to be completed over, perhaps, the next month. The Master Production Scheduling II function breaks this large amount of work into viable subsets that can be manufactured within the constraints and broadly in line with the aims of the automated manufacturing system over a much shorter time period (perhaps 8 hours).

On Line Scheduling - this takes the viable subset of work from the Master Production Scheduling II function and breaks this down into loading specific machines or workstations

Resources Checked - the resources essential to the completion of a job (for example, raw materials, tooling components) are checked to confirm availability. If they are not available then they are reordered (with higher priority)

Workstation/Cell Scheduled - detailed schedules are produced. For very large automated manufacturing systems then this may be done at two stages. Firstly the cells are scheduled and then each workstation is scheduled.

Generate Instructions to Machines - the process planning data base is used to obtain the NC part programs and these are downloaded to the machines. Detailed sequence instructions from the workstation/cell schedule function are then used to drive the equipment.

Problem and error recovery - in all likelihood the equipment operation will differ in some respects from that planned. A problem and error recovery function can be used to alleviate any problem caused.

Two major scheduling and control functions are Master Production Scheduling II and On-Line Scheduling and these sections are discussed later in more detail.

4. CONTROL HIERARCHIES

The functions briefly described in the preceeding section can be mapped onto a varying number of scheduling and control levels. At one extreme, each function can occupy a separate level. At the other extreme all the functions could be lumped together into a single control level.

One major approach has been that of the National Bureau of Standards in the Automated Manufacturing Research Facility (AMRF) [see Jones and McLean (1984), Furlani et al. (1983)]. Five levels have been proposed.

At the top is the Facility level which includes Process Planning, Production Management (including long range schedules) and Information Management (including links to financial and other administrative functions).

Below this is the Shop level which manages the coordination of resources and jobs on the shop floor. The processes involved at this level include the grouping of jobs into part batches using a Group Technology (GT) classification scheme. The concept of a virtual manufacturing cell is introduced at this stage. These virtual manufacturing cells comprise machines which are grouped together in a dynamic fashion, that is the configuration and number of virtual manufacturing cells varies with time. Besides job groups and virtual manufacturing cell configuration, the tasks at this shop level include allocating tooling, jigs/fixtures and materials to specific workstation/job combinations. These activities at the shop level are re-evaluated on the basis of feedback from the cell level and on changes in requirements etc. from the facility level.

Below the shop level is the cell level, where the cell control system schedules and controls jobs through the cell. These jobs have already been divided into groups so that the jobs allocated to each cell are somewhat similar. Also involved in scheduling and controlling the jobs is the scheduling of material handling and tooling within the cell.

The next lowest level is the workstation level which consists of coordinating the activities of the AMRF workstation which is taken to typically consist of a robot, a machine tool, a material storage buffer and a control computer. The workstation controller then arranges the sequencing of operations so as to complete the jobs allocated to the cell control system.

The lowest level of the planning and control hierarchy is the equipment level which consists of the controller for individual resources such as machine tools, robots or material handlers.

A second approach to the number of levels is that of Computer Aided Manufacturing International Inc. (CAM-I). This is an organization dealing with the design and implementation of computer technology to manufacturing. It operates on a consortium basis in that individual companies join and then pay a further subscription to join one or more of a number of programs which carry our activities in a particular area. One particular program is the Factory Management Program which is concerned with the design and implementation of a factory management system to efficiently manage production.

The Factory Management Program has designed a computer hierarchy of control using four levels:

Factory Control System - this is at the top level and is concerned with top level factory management considering such aspects as determining end item requirements, determining product sutructure definitions (Process planning)

and determining individual shop capacities and capabilities.

Job Shop Level - this level is the next below the Factory level and takes commands from the Factory level to determine commands for the work center levels. Included in this would be the taking of end item production and exploding this into processing operations. Having done this the shop order events are scheduled.

Work Center Level - this level takes the commands from the job shop level and generates detailed task requirements. The task events are then scheduled and commands for these tasks are passed to the next level - unit/resource level.

Unit/resource Level - the tasks from the work center level are broken into subtasks and these subtasks are carried out.

Associated with each level there is also a feedback mechanism whereby occurance of events is fed back to the next higher level. This procedure leads to a relatively decentralized structure whereby decision making is made at the lowest possible level commensurate with overall efficiency. Control of each level resides at the next highest level and this next highest level issues commands to the next lowest level. This lower level gives feedback on its current status to the next highest level to facilitate decision making at that level. As we progress down the hierarchy the planning horizon shortens. At the top factory level, we may be concerned with planning horizons of perhaps months whereas at the unit/resource level then the planning horizon may only be measured in seconds or minutes.

The present developmental status of the CAM-I AFMS is that a comprehensive data flow model has been completed together with a data dictionary. No implementation of these models has as yet been carried out and this stage awaits further developments.

A four level hierarchy has also been proposed by O'Grady (1986). These levels consist of:

Factory - containing the major corporate computer systems such as process planning, computer aided design, master production scheduling I, materials requirements planning and the financial systems

Shop - containing the master production scheduling II and on-line scheduling systems

Cell - this level schedules and controls cell activity

Equipment - this level controls the equipment.

It should again be stressed that rather than concentrating on and discussing the number of levels, what is probably of more importance are the functions that need to be carried out (for an example, see Figure 1). These functions can be mapped onto a wide number of layers in a hierarchy.

6. MASTER PRODUCTION SCHEDULING II (MPS II)

The MPS II function level is concerned with extracting from the "work-to" list (obtained from the Factory Materials Requirements Planning (MRP) system) a viable portion of work to be done in a specified time period. The "work-to" list may contain work to be done over perhaps the next month and the MPS II breaks that down into work to be done in a much shorter time period. The average operation times are important in deciding how long this much shorter time period should be. With longer operation times then the time period considered by the MPS II function could be longer and where operation times are very short then the MPS II may only be concerned with time periods of perhaps an hour.

Superficially the MPS II function is similar to the MPS I function that precedes the MRP calculations of net requirements as indicated in Fig. 1. However, the two MPS functions, MPS I and MPS II, do differ in two significant respects:

1. The time periods considered are different - MPS I may be for weekly time periods whereas MPS II may be for time periods of a few hours.

2. MPS II may have to consider engineering details such as tools or jig/fixture supply.

There are analytical methods of determining an effective and viable MPS I [see O'Grady and Menon (1985), O'Grady (1986)] and these approaches are capable to a greater or lesser degree of giving effective solutions for the MPS I function for conventional manufacturing. Obviously any analytical solution will need to be treated with care to ensure that all important factors have been included in the derivation of the solution. Given this general warning, the conventional approaches each have one or more of a number of drawbacks when applied to MPS II in a typical automated manufacturing system. The first drawback concerns the need to consider tooling and the provision of jigs/fixtures as a major constraint. Although the virtual capacity tool magazine has become technically viable, it is the exception rather than the general case and a finite capacity tool machine is the norm at present. Carrie et al. (1984) cite the example of a major automated manufacturing system whre a product mix of just seven part types requires all 100 tool magazine slots at certain machines. In a practical automated manufacturing system there is also likely to be a limit on the number of particular

jigs/fixtures available, because of the high cost of producing sophisticated jigs/fixtures, and therefore there are likely to be constraints on the use of such jigs/fixtures. This means that the MPS II function must not only take into account such factors as machine capacity as a constraint but also such other factors as the provision of tooling and jigs/fixtures.

The second drawback concerns the nature of the constraints themselves. Most approaches would consider the constraints as being "hard", i.e. no overshoot of the constraint is possible. In many facilities this does not adequately reflect the decision making process; for example the tooling required can exceed the machine capacity if the drawback of the time taken to physically change the tooling can be accomodated presumably with some reduction in system performance. Any analytical procedure must therefore be capable of including constraints as both 'hard' and as 'soft'.

The third drawback concerns the goals of the automated manufacturing system; often there are a wide range of sometimes partly conflicting goals, for example, the goals of high machine utilization and low work in progress levels. There is also likely to be a wide difference in goals from installation to installation. There is therefore a requirement that the methods can handle this wider variation in goals.

The fourth and final drawback to many of the analytical approaches is that they often produce aggregate solutions which have to be then disaggregated to obtain the desired production rate for each product.

Some authors have considered the particular requirements of automated manufacturing systems: Stecke (1983), for example, has constructed a non-linear integer programming model for a particular automated manufacturing system and she was able to obtain some solutions using a standard mixed integer programming system. This approach however, does suffer from the usual problem associated with mixed integer programming namely that of high computation times. Queueing theory approaches have been considered by several researchers including Solberg (1976), Buzacott and Shanthikumar (1980), and Yao (1983). These Queueing Theory approaches can provide some useful results in the design phase of FMS but their usefulness in providing detailed operational solutions has to be doubted.

O'Grady and Menon (1985) have proposed an adapted goal programming approach which overcomes many of the disadvantages of the conventional approaches.

The O'Grady and Menon (1985) approach is orientated to a general automated manufacturing system to include a wide variety of constraints including both 'hard' and 'soft' constraints. In

addition multiple goals are readily incorporated. The approach considers such aspects as:

* tooling requirements
* linked groups of orders
* machine capacity
* alternative routes
* due date considerations
* volume of work in progress
* the expediting of certain orders.

O'Grady and Menon (1985) give a detailed account of how each of the above aspects can be formulated within their framework, the essential philosophy being that users can select to the users' own particular requirements individual modules.

The computing times associated with this approach, however, mean that it is only suitable for use when sufficient time allows. At some future point in time, computing power may have risen to such an extent that the approach can be used in real time. For the forseeable future though, the approach may have to be augmented by a 'mailbox' facility which contains some rules/heuristics which can readily determine a reasonable MPS II without a significant computing time delay. The production Logistics and Timings Organizer (PLATO) system provides such an environment [see O'Grady and Brightman (1986) and O'Grady et al (1986)].

Whatever approach is used, the output of the MPS II function is a work load for the time period under consideration. This now passes to the next stage, that of On-Line Scheduling.

7. ON-LINE SCHEDULING

The On-Line Scheduling function is concerned with taking the MPS II output which is a viable work load for the particular planning period and translating this into instructions to each cell/workstation taking into account the status of the automated manufacturing system and the overall requirements. We can stress overall; we are concerned with the overall performance of the automated manufacturing system not just with optimizing one particular cell since this may give poor overall system performance.

The degree of detail required in the instruction to each cell/workstation will vary with the ability of the cell/workstation as well as what degree decentralized control is favored. Where centralized control is required and/or there is limited decision making ability at the cell/workstation level, the cell/workstation requires relatively detailed instructions. Conversely, where decentralized control is required and there is a reasonable decision making ability at the cell/workstation

level then the level of detail required is much less and perhaps instead broad goals for the cell to achieve replace detailed instructions.

The major practical approach to On-Line Scheduling was that of a simple fixed heuristic operating on the queue of jobs [see Panwalker and Iskander (1977)] whereby a priority is allocated to each job in the queue. The job with the highest priority is selected when the machine becomes vacant. The fixed heuristic approach has the advantage of ease of computation but it has the disadvantage that the solution obtained may be poor when applied to a wide variation of system specifications. The requirements of the On-Line Scheduling function within an automated manufacturing system environment are such that good performance is required across a wide range of operating conditions. Optimal approaches in general require large computing resources and a heuristic approach which adapts to a particular automated manufacturing system to give good results across a wide range of systems operations seems to be the best compromise in terms of computing times and quality of the solution. Amongst the first to propose the use of adaptive heuristics has been Fischer and Thompson (1963) with some later work done by Hershauer and Ebert (1975). The approaches used by these authors however suffer from the disadvantage that neither uses a unified format and consequently the methods have to be considerably altered for each application, making their use difficult in practice. The approach of O'Grady and Harrison (1985) is to use a heuristic that not only adapts to the manufacturing system but also is one that is expressed in a unified format. This approach, termed Search Sequencing, uses a priority rule operating on the queue of waiting jobs at each machine or process. The coefficients in the priority rule adapt (using a search routine) to a particular manufacturing system. The search technique used by O'Grady and Harrison (1985) is a modified version of the Hooke-Jeeves (1961) pattern search routine. The modifications are to reduce the risk of the search becoming trapped in local minima. Other search techniques could be used however, and their efficiency may well suit particular automated manufacturing system environments. Advantages of this approach include the adaptation to particular manufacturing systems and particular performance measures leading to considerable improvements over fixed heuristics. In addition, the approach can be simplified or made more detailed depending on the users requirements. Tests on practical manufacturing systems indicate a good performance over a wide range of manufacturing systems.

8. CONCLUSION

The problem of scheduling and controlling the vast majority of manufacturing systems is tremendously complex. This complexity increases when automated manufacturing systems are

considered. This paper has given both the background and some possible approaches to scheduling and control of automated manufacturing systems.

The factors affecting scheduling and control of automated manufacturing systems have been indicated and those factors which increase the complexity of the problem include:

1. Shorter manufacturing lead times
2. Engineering details need to be considered
3. Greater emphasis is placed on system utilization
4. Need to integrate with existing software system
5. Detailed instructions need to be generated

The use of automated data collection can ease the problem somewhat by the provision of higher volumes of high accuracy, timely data that such data collection systems can provide.

The aim of a scheduling and control system for automated manufacturing systems is to take the broad system goals and translate these into specific instructions for each item of equipment responding to problems as and when they occur. This paper has described an example of the major functions of such a scheduling and control system. Viable and effective methodologies to the most important functions have been described.

9. REFERENCES

Buzacott, J. A. and Shanthikumar, J. G. (1980), "Models for Understanding Flexible Manufacturing Systems," American Institute Industrial Engineers Transactions, 12, 4.

Carrie, A. S., Adhami, E., Stephens, A., and Murdock, I. C. (1984), "Introducing a Flexible Manufacturing System," International Journal of Production Research, 22, 6.

Furlani, C. M., Kent, E. W., Bloom, H. and McLean, C. R. (1983), "The Automated Manufacturing Research Facility of the National Bureau of Standards," Proc. Computer Simulation Conference (Vancouver, B.C.)

Hershauer, J. C. and Ebert, R. J. (1975), "Search and simulation selection of a job-shop sequencing rule," Management Science, 21, 833.

Hooke, R. and Jeeves, T. A. (1961), "Direct search solution of numerical and statistical problems," Journal of the Association of Computing Machinery, 8, 212.

Jones, A. T. and McLean, C. R. (1984), "A Cell Control System for the AMRF," ASME Conf., August.

O'Grady, P. J. (1986), "Controlling Automated Manufacturing Systems," Kogan-Page.

O'Grady, P. J. and Brightman, M. (1986), "The Application of Artificial Intelligence to Production Planning and Control," Working Paper 86/02, Dept. Ind. Engr., North Carolina State University, Raleigh, NC 27695.

O'Grady, P. J., Bao, H., and Lee, K. H., "The structure of an intelligent cell controller for automated manufacturing systems," Working Paper 86/03, Dept. Ind. Engr., North Carolina State University, Raleigh, NC 27695.

O'Grady, P. J. and Harrison, C. (1985), "A general search sequencing rule for job-shop sequencing," International Journal of Production Research, 23, 5.

O'Grady, P. J. and Menon, U. (1985), "A multiple criteria approach for production planning of automated manufacturing," Engineering Optimization, 8, 3.

O'Grady, P. J. and Menon U. (1986), "Master production scheduling for a flexible manufacturing system," to be published.

Panwalker, S. S. and Iskander, W. (1977), "A survey of scheduling rules," Operations Research, 25, 1.

Solberg, J. J. (1976), "Optimal design and control of computerized manufacturing systems," American Institute of Industrial Engineers Conference (Boston), 138.

Yao, D. D. W. (1983), "Queueing models of flexible manufacturing systems," Ph.D. Thesis, University of Toronto.

THE INTEGRATION OF PLANNING, SCHEDULING, AND CONTROL FOR AUTOMATED MANUFACTURING

THOMAS E. BAKER and DWIGHT E. COLLINS
Chesapeake Decision Sciences, Inc.
200 South Street, New Providence, NJ 07974

Introduction

This paper discusses a hierarchical approach to the integration of the planning, scheduling, and control (P/S/C) functions found in the automated manufacturing environment. We first define what we mean by P/S/C, indicate the origins of the ideas used in our approach, and describe our efforts to make these ideas work in practice.

The literature contains numerous theoretical discussions of hierarchical approaches to P/S/C. While this paper makes reference to some of the previously published underlying concepts, it focuses primarily on the considerations, steps, and problems involved in actually implementing a hierarchical framework for P/S/C.

One might best interpret our implementation of the hierarchy in terms of a data base-oriented application of generalized math programming principles. The core of the framework is a data base which is a single source of information for the three business functions of P/S/C. The data base is structured such that, for example, data required by both the planning function and scheduling function exists in a format that can be used effectively by both functions. A variety of algorithms is available to the user to invoke within this framework in pursuit of specialized or local objectives. The choice of algorithms to use, the sequence in which they are used, and the rules by which they are used are determined by the user's overall objectives and environment.

Both formal and informal applications of primal and dual decomposition are used to generate primal (quantity target) and dual (cost or economic target) information for the system-wide data base. This primal and dual information is used in turn by the algorithms in the hierarchy to insure that the local actions taken by specialized algorithms are compatible with the overall goals of the entire system.

We use decomposition techniques to integrate not only the different levels of the P/S/C hierarchy but also the actions of various algorithms within a given level. For example, we apply these principles to enable us to solve sequencing, lot sizing, and resource allocation problems independently and yet have the results mesh in an integrated manner.

The P/S/C Hierarchy

The diagram in Figure 1 represents our hierarchy in terms of the business functions which are performed at each level. Systems Operation Planning is concerned primarily with longer term, multi-plant planning. The Plant Operations Planning Function is concerned with the medium-term planning of a single plant. The Scheduling Function has been separated from the Operations Planning Function to distinguish the considerations of the sequencing of operations and the timing of events which take place at the scheduling level. Both the Plant Operations Planning Function and the Scheduling Function interact heavily with the Material Requirements Planning Function. Finally, both the Scheduling Function and the Material Requirements Function interact with Job Control and Inventory Control.

Hierarchical Approach

The structure of our framework for modeling the functional hierarchy reflects the following objectives:

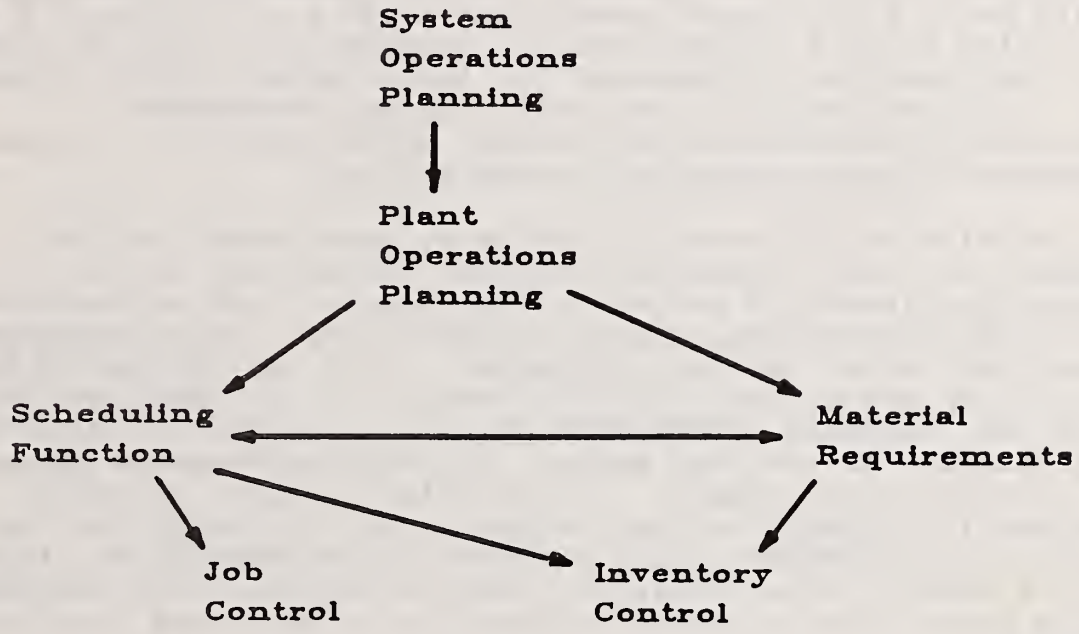
- Avoid large models based on a single technique,
- Develop small, specialized models at each level incorporating those techniques which are appropriate at each level, and
- Develop a framework for information transfer between levels.

At first glance, the approach may seem counter-intuitive; we propose to integrate the hierarchy by breaking it apart into smaller components. In a sense this is true. The approach has two key properties:

- The first two objectives recognize how many firms maintain data and do analysis within their P/S/C functions today, given typical organization structures. In particular, the approach is consistent with the fact that planning is done in one organization, scheduling in another, etc. Furthermore, it offers a mode of use that is compatible with these organizational realities.
- At the same time, it offers an approach to dealing with the key problem of information transfer between levels. This problem is well documented. For example, Odrey and Nagel (1986) point out how many of today's data base structures for supporting decision making at the operational level are unable to provide meaningful information in support of planning level decisions.

Figure 1

MANUFACTURING INDUSTRY LOGISTICS



Decomposition Methods

As mentioned in the Introduction, we apply formal and informal decomposition techniques from mathematical programming theory to generate and pass information between different levels in the hierarchy. For example, consider the diagram shown in Figure 2, depicting the interface between a planning function (P) and a scheduling function (S). In a real sense, we might think of the scheduling problem as a subproblem of the planning problem in that the planning problem may encompass many similar scheduling problems with its larger scope and longer-term view.

As shown in Figure 2, suppose that the planning function determines that the scheduling function should produce A parts. The scheduling function determines that it should product A' parts, not A parts. How do we reconcile the difference so that the overall plan is consistent? One method would be to fix a target production level A = A' in both P and S. Solve P and S independently. Determine the cost to produce (or the incentive to produce) the target level in both P and S. If the economics do not match, adjust the target accordingly and repeat the process. This approach is called Resource-Directed or Primal Decomposition.

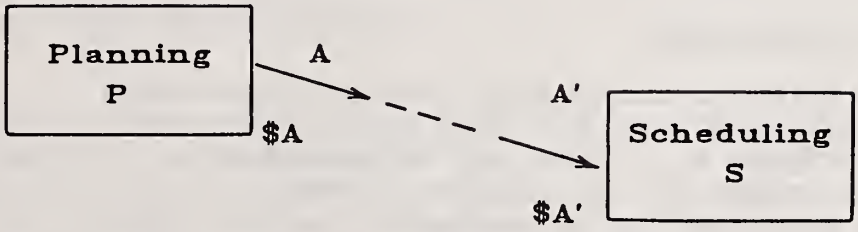
An alternate approach would be to pass target economic values, $A = $A', which are adjusted iteratively if the production levels, A and A', do not match. Such an approach is called Price-Directed or Dual Decomposition. Both approaches have their strengths and weaknesses. Both approaches are used in the implementation of our hierarchy. It is important to note that economic information plays an important role in both approaches. In the Dual method, economic information becomes the coordinating signals. In the Primal method, economic information appears in the feedback loop. In industry, the failure of successful P/S/C integration can usually be traced to the absence of a consistent base of manufacturing economics which would allow economic information to be passed from level to level.

Integration Framework

As we have indicated, the key to successful P/S/C integration is an economic base which is consistent for all levels in the hierarchy. Unfortunately, most existing cost accounting systems seem to obscure true manufacturing economics. Reasonable choices for an economic base are total controllable (as opposed to fixed cost allocation) manufacturing cost and base load manufacturing cost. However, the choice of economic base is probably less important than having one that is consistent across all levels in the hierarchy.

Figure 2

Decomposition Methods



Primal	Dual
1. Choose target = A = A'	1. Choose price = $A = $A'
2. Solve P & S	2. Solve P & S
3. Compare prices	3. Compare production
If $A ≠ $A' return to 1.	If A ≠ A' return to 1.

Once we've chosen an economic base, the next task in establishing our Integration Framework is the development of a function-wide data base to support all P/S/C functions. Although we will have little to say about data base development in this paper, we recognize that this probably represents the most difficult task in P/S/C integration.

After the function-wide data base has been established, we can begin thinking about the implementation of algorithms which perform operations on the data base. This approach is diametrically opposed to the traditional operations research approach of developing self-contained models which are subsequently beaten senseless by formal optimization techniques. Our integration framework contains no self-contained models, only a data base. Many of the traditional optimization techniques are applied, but in a form which is an integral part of the function-wide data base.

Historical Perspective

The historical relationship between algorithmic (optimization) expertise and data base (environment) expertise is shown in Figure 3. There may be some overlap of algorithmic and data base expertise in industry although the point is debatable. Such overlap in academia does not appear to exist. Historically, the academic algorithmic research in planning and scheduling has drifted off to complexity theory. Meanwhile, industrial data base experts, feeling that the academic community has not been solving their problems, have embraced data base intensive approaches to P/S/C like MRP.

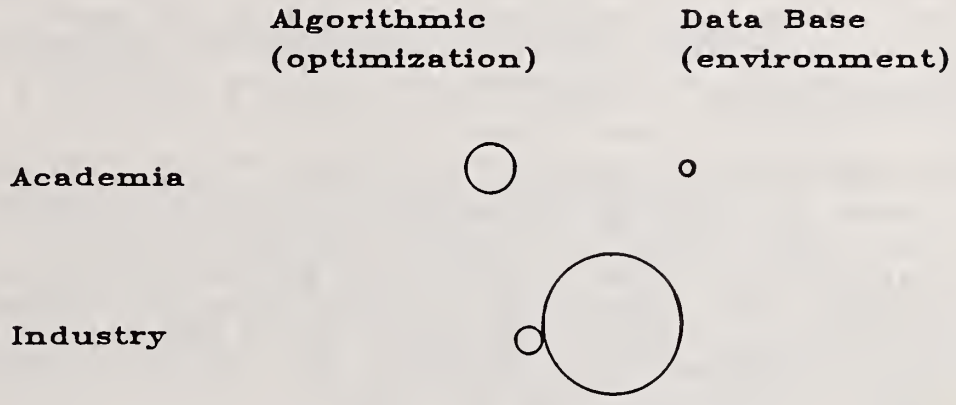
We propose to imbed the best of the academic algorithmic developments in the industrial data base environment.

Origins

Before proceeding with the presentation of our hierarchical framework, it is useful to note some of the origins of the basic ideas upon which we rely. Many of the key concepts of the hierarchical approach can be traced back to Mesarovic (1970) and Lefkowitz (1977) whose work on hierarchical systems theory was responsible for many subsequent developments in multi-level control in the process industry. Their research was concerned with the generation and transfer of information (primal and dual) up and down the hierarchy. It is interesting that this early reference was related to the process industry. As manufacturing becomes more flexible and automated, manufacturing industry problems tend to approach well-known process industry problems. Hence, this reference is highly relevant.

Figure 3

Venn Diagram Showing Overlap of Expertise



In the area of manufacturing production scheduling, one of the earliest important references is, of course, the work by Hax and Meal (1975) on the hierarchical integration of production planning and scheduling. This work was responsible for the flood of hierarchical planning publications which have eminated from MIT over the last ten years. However, it is interesting to note that this original work considered solely the top-down flow of information comprised only of primal signals in the form of constraints.

While others were working on theory, Jaikumar (1974) was implementing a similar planning and scheduling hierarchy in practice. His hierarchy added costs to the constraint information flowing from the planning level to the scheduling level but made no provision for feedback from scheduling back to planning.

A fairly recent paper by Graves (1982) revisits the Hax and Meal hierarchy and applies Lagrangean Relaxation to decouple the planning and scheduling problems. With this approach, the planning and scheduling functions receive Lagrangean multipliers (economic or dual information) from a master coordination function and develop production levels (physical or primal information) which are fed back to the master coordination function. As will be seen in the following sections, much of our implementation follows the spirit of the Lagrangean approach.

Other researchers, Maxwell, Muckstadt, Thomas, VanderEecken (1983), have examined the hierarchical approach in terms of the individual tasks (e.g., lot size and protection stock determination) which must be performed at the various levels in the hierarchy. The separation of these tasks and the analysis of the information required by each task is important to our hierarchical approach which considers all tasks as separate operations on a function-wide data base.

Planning, Scheduling, and Control Tasks

A useful characterization of the P/S/C tasks as viewed by these and other researchers is shown below:

Planning/Scheduling Hierarchy (Graves, Maxwell, Yano, et.al.)

Manufacturing Production Planning
* Aggregate Production Smoothing
* Lot Sizing and Timing
  Reorder Intervals

Manufacturing Flow Planning
 * Resource Allocation
 Planned Lead Times
 Protection Stock

Manufacturing Scheduling
 * Detailed Sequencing
 * Inventory Control
 Dynamic Rescheduling

The implementation of our hierarchy follows, to a large degree, the classification of tasks shown above. Note, however, that we are concerned not only with information flow between levels, but also with information flow between individual tasks and the algorithms which perform them.

The implementation of the five tasks marked above with asterisks (*) will be discussed below in some detail. But first, let us demonstrate the algorithm/data base interaction with a simple example.

Algorithm/Data Base Interaction -- A Simple Example

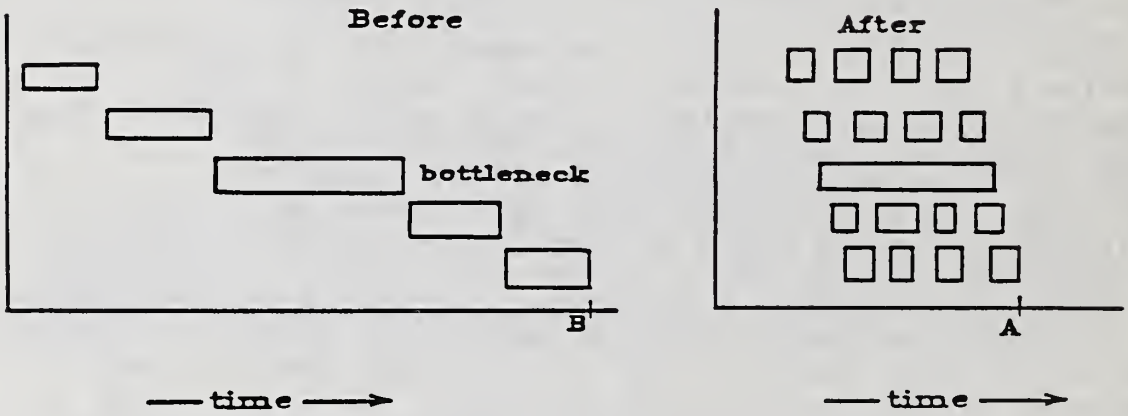
Figure 4 provides a graphic representation of the before and after solution of a well-publicized example of the scheduling philosophy required to maximize the throughput of a plant. The Gantt chart on the left represents the "traditional" method of scheduling a job through the plant--each job step operation (process batch) is run to completion before the parts are passed to the next operation. The Gantt chart on the right is the result of (i) identifying the bottleneck resource, (ii) maintaining long process batches for operations on that bottleneck resource in order to minimize time lost on that resource, and (iii) running short process batches on non-bottleneck resources which, in effect, feed the bottleneck. The result is that the job now finishes in less time, minimizing work-in-process (WIP) and increasing operations flexibility.

If our goal is to maximize factory throughput, this bottleneck analogy/model offers a valid solution approach. (For the moment, let's put aside considerations of maximizing profit or of positioning our plant for seasonal inventories or other external conditions.)

But let us ask the following question: From the point of view of our hierarchical approach, which algorithms and what data base structure are required to reach the same conclusion under the same conditions? The answer is reassuringly simple.

Figure 4

Debottlenecking for Throughput Maximization



All we need, to produce the same result, is a data base which understands the value of resources over time and the Economic Order Quantity (EOQ) formula for determining lot sizes (process batches).

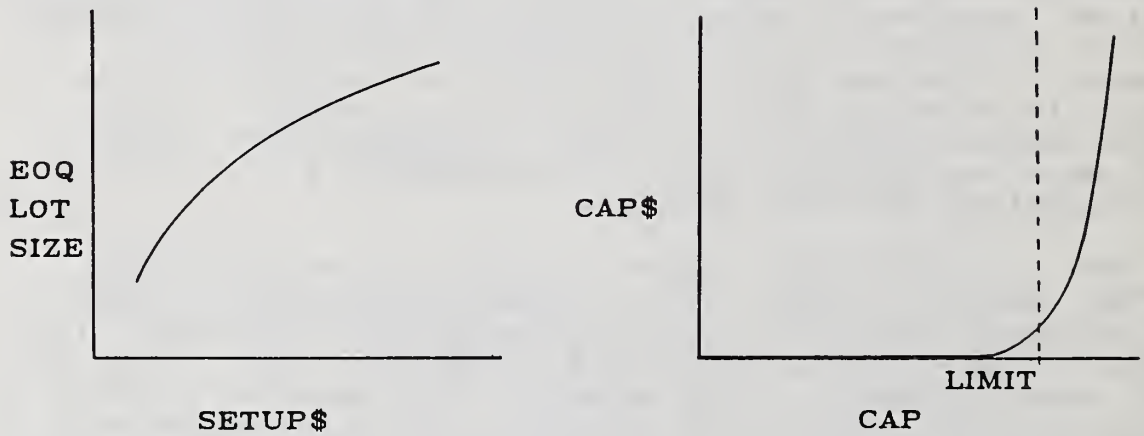
Figure 5 provides a graphical representation of the main argument. The graph on the left indicates that, all other things being equal, EOQ lot size increases as a function of setup cost. The graph on the right represents the economic value of a resource for a specified period of time. Assume for a moment that the capacity utilization, CAP, of a resource has to be less than some specified limit (e.g., a particular machine has only eight hours capacity in one shift.) If the capacity utilization on that resource is near or above its limit, then that resource is limiting the throughput of the plant (again, assuming that throughput maximization is our goal) and the capacity takes on some economic value, CAP$. This might be, for example, the opportunity cost of not producing additional units of output from the factory per unit of time, relative to producing output in another more expensive factory or relative to lost revenue from sales foregone. However, if the capacity utilization on a resource is below its limit, its economic value, CAP$, takes on a value of zero. Those readers familiar with linear programming (LP) concepts will note that our CAP$ value is analogous to a dual value for an inequality row in an LP model.

The only remaining linkage to define is the setup cost, SETUP$, which is simply, (i) the direct dollar cost, DIRECT$, of performing the setup plus (ii) the resource time lost, TIME LOST, during the setup multiplied by the time value, CAP$, of the resource capacity being consumed. The equation is shown in Figure 5. Thus, bottleneck resources take on a high value for their capacity, resulting in larger setup costs and therefore larger process batches (EOQ lot sizes). Non-bottleneck resources have a zero value for their capacity, producing smaller setup costs and hence smaller process batches. Thus, a data base containing economic (dual) values (like CAP$) for finite resource capacity can transform a simple (myopic) EOQ formula into a tool which makes reasonable decisions in a global sense.

This example serves only to illustrate the spirit of our approach. As we will see below, the EOQ formula is a little too simple to be of use in complex scheduling problems.

Figure 5

Effect of Capacity
Dual Values On Lot Size



CAP ≤ LIMIT ➡ CAP$

SETUP$ = DIRECT$ + TIME LOST * CAP$

Bottleneck ➡ Increase In Lot Size

Constraints and Boundary Conditions

Figure 6 represents a view of the constraints and boundary conditions which make planning/scheduling problems difficult. The variables in upper case can be thought of as given or fixed. Values of the lower case variables are determined by the model. For example, Requirements (parts or tools required as input to produce end items within the schedule) must be less than or equal to the expected RECEIPTS within the order lead time for the required parts or tools. The Production of end items must be greater or equal to the DEMAND.

As several researchers have observed, planning/scheduling problems are two-point boundary problems--going from an INITIAL INVENTORY to a Final Inventory greater than or equal to a TARGET while satisfying the conditions:

$$\begin{aligned} \text{Requirements} &\leq \text{RECEIPTS} \\ \text{Utilization} &\leq \text{CAPACITY} \\ \text{Production} &\geq \text{DEMAND} \end{aligned}$$

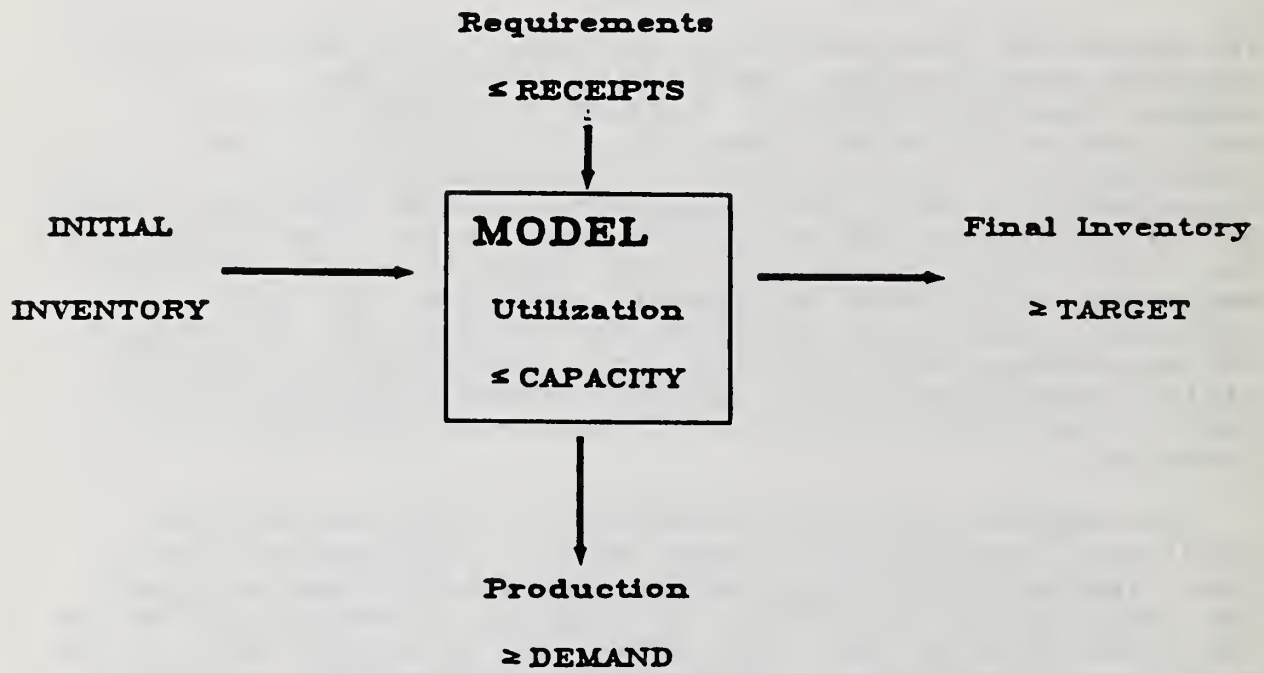
throughout the time horizon of the model. Like many two-point boundary value problems, the problem above may be computationally intractable in practice (unless there is sufficient slack in the constraints so as to render them unnecessary). In recognition of this computational intractability, most planning/scheduling algorithms ignore some of the constraints and attempt to generate solutions which meet the conditions of one or two other constraints. For example, MRP starts with DEMAND and TARGET inventories and works backward to produce Requirements and Utilization. In doing so, MRP ignores the constraints related to RECEIPTS and CAPACITY. Finite forward scheduling algorithms start with RECEIPTS and CAPACITY and work forward to determine Production and Final Inventory.

The important distinction between our approach and the traditional planning and scheduling system implementations is that, instead of ignoring complicating constraints, we relax the complicating constraints and generate Lagrangean prices for the constraints so that their effect can be incorporated in the traditional algorithms. In the following sections, we describe five of our implementations of some of the standard OR-based techniques:

Just-In-Time MRP
Capacity Balancing
Lot Size/Frequency Determination
Resource Allocation
Sequencing

Figure 6

Constraints & Boundary Conditions



Material Requirements Planning (MRP)

Our implementation of MRP logic can be used at both the production planning and the flow planning levels. Just-In-Time flow planning can be approached through the specification of adjustable lot sizes and minimal manufacturing lead times. MRP relaxes the receipt of requirements (Requirements < RECEIPTS) in order to generate a flow plan. Capacity utilization (Utilization < CAPACITY) can optionally be treated as a hard constraint or relaxed in infinite capacity mode. Under the infinite capacity mode, the MRP routine generates Lagrangean values for capacity utilization in much the same manner as that indicated in our debottlenecking example shown above. These values are used by other algorithms to reflect the influence of the finite capacity constraint.

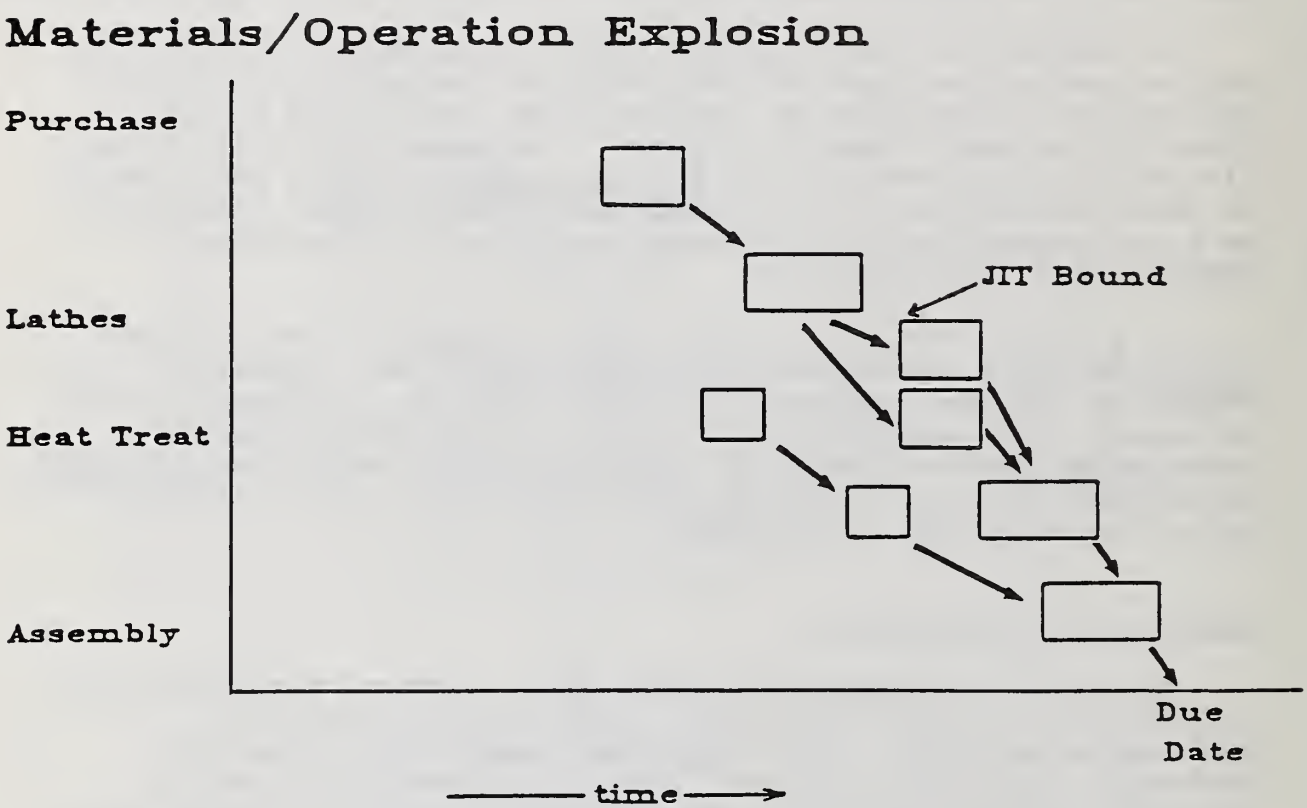
In addition to the economic (dual) information contained in Lagrangian values, the MRP routine also creates bound (primal) information in the form of Just-In-Time job release bounds. These JIT bounds, shown in Figure 7, represent the latest start time which a job step can have if the corresponding end item is to meet its due date. JIT bounds are used by other algorithms as a representation of the demand constraint (Production $\geq$ DEMAND) on a job step level.

In a sense, the MRP routine can be thought of as a greedy algorithm which schedules activities at the last possible moment on the best possible facility. While it is usually necessary to do quite a bit of work on this type of greedy solution to make it feasible, the initial MRP solution (and the associated auxiliary information) represents a good starting point for the other algorithms.

Capacity Balancing (BAL)

To the extent that the factory has alternate facilities for performing the same operations, there may be a facility preference order for each operation. Algorithms like MRP generally chose the facility of first preference for each operation during the materials/operation explosion. Long term capacity balancing entails reordering the facility preference list in order to balance loads across facilities and to cluster similar operations on the same facilities in order to ultimately reduce setup costs.

Figure 7


Materials/Operation Explosion
Purchase
Lathes
Heat Treat
Assembly
JIT Bound
Due
Date
time

The longer term preference clustering of similar operations across parallel facilities requires that we make an approximation of the setup cost among operations assigned to a facility during a given production period. (Setups are sequence dependent and we haven't yet determined detailed sequences for the operations on each machine.) In the BAL routine, we use a production period approximation of setup cost similar to the two-phase vehicle routing approach of Tyagi (1968). In the multiple vehicle routing problem, Tyagi first approximated travel costs from points to routes in order to cluster points to be visited to the proposed vehicle routes. Then, in the second phase of the algorithm, the points within each route were sequenced. It may appear strange in a paper on production scheduling to reference a paper on vehicle routing. However, as the discerning reader will note, the parallel machine loading problem and the multiple vehicle routing problem are similar if not identical from the mathematical and algorithmic points of view.

In the BAL routine, operations are clustered to facilities based on the capacity available on the facility and the similarity (from a setup standpoint) of an operation with those operations already assigned to that facility. Detailed sequencing of operations on each facility is handled by the SEQ routine as described below.

The BAL routine would be run infrequently; prompted by changes in demand mix or capacity availability.

Lot Size/Frequency Determination (LOTS)

During the MRP calculations, lots sizes (process batches) may be of fixed size or may be adjustable (within a minimum and a maximum) to the demand for that operation during a given time period. However, once a production plan has been established, it may be worthwhile to return to question of lot size/frequency determination on an optimal basis.

We follow the single product, optimal lot size/frequency procedure proposed by Wagner and Whitin (1958). This dynamic programming-based approach uses sequence-dependent setup costs and the internal demand generated for a single product for the current schedule to produce the optimal lot size/frequency for a single product. We have extended the Wagner-Whitin procedure to include a consideration of the finite capacity dual (Lagrangean) values discussed above.

This additional dual information allows the procedure to optimally determine the timing of the lot size to avoid periods of over-utilized capacity. For example, the dotted lines in Figure 8 might indicate the timing of optimal lot sizes as determined by the original Wagner-Whitin algorithm. However, the second lot falls in a period of high capacity utilization. Since our implementation of the algorithm considers setup costs to include direct costs plus the value of the time lost on the facility, the setup cost during this period would take on a higher value which would move the second lot earlier to the position shown in the diagram.

Capacity Allocation (CAP)

In flexible manufacturing, the concept of a static, identifiable bottleneck tends to break down. Bottlenecks to production tend to move throughout the factory at different stages of the job cycle and with changes in product mix. Quite often, the capacity allocation decisions made at the production planning level by BAL (capacity balancing) have to be remade at the flow planning level on a dynamic, operation-by-operation basis.

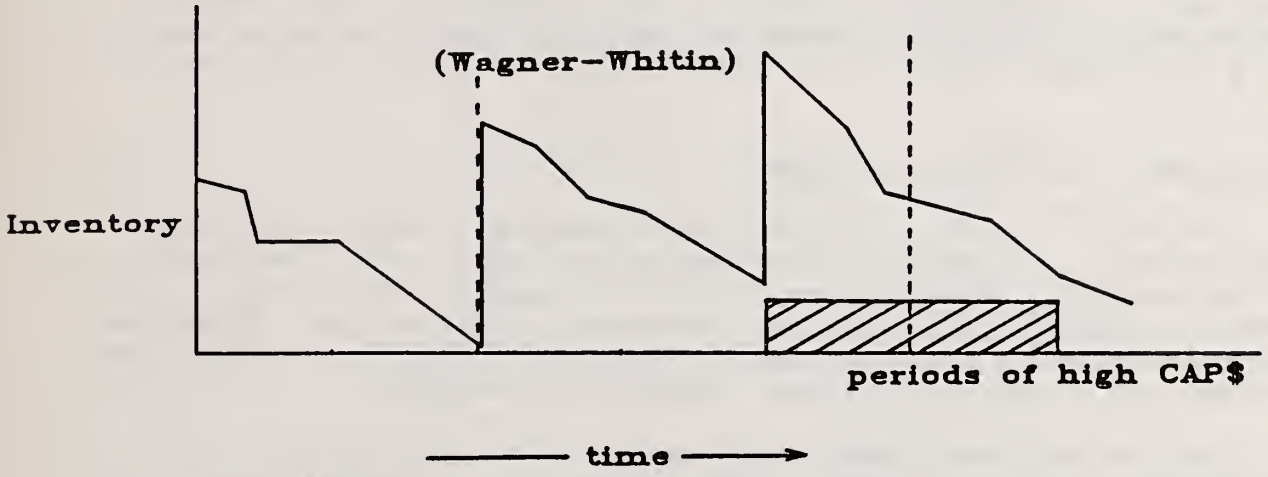
The CAP routine focuses on the dynamic reallocation of operations across parallel machines and over time. The production period approximation of setup costs, the capacity dual values, and facility cost differentials are used to optimize the reallocation of scheduled operations. A network-based approach is used to decide whether bottlenecked operations should be reallocated to other facilities and/or earlier in time.

Operations Sequencing (SEQ)

The SEQ routine uses an adaptation of Lin's k-opt method (1965) for resequencing a given string of scheduled activities. This technique attempts to minimize the sum of setup costs for the entire string. In our implementation, the size of the string is limited by the WIP constraints involved in multi-stage processing and the displacement of scheduled activities is limited by the just-in-time bounds created by the MRP routine described previously. As with the LOTS routine, setup costs include direct setup costs and the value of the time lost on the setup facility. Thus the k-opt method is augmented with both primal and dual information generated by other algorithms.

Figure 8

Effect of Capacity Dual Values on Lot Size/Timing



Scheduling/Control Interface

As we work our way down the planning/scheduling/control hierarchy, we reach the interface between scheduling and factory control. This interface is extremely critical due to the fact that the scheduling function in the automated manufacturing environment must approach the time frame and the detailed considerations found in real-time control.

As in the higher level interfaces, we have the choice of whether to pass primal (targets) or dual (economics) information. The diagram in Figure 9 illustrates two of the configurations which we might consider. Assume that we are going to use some sort of dispatching logic to generate a final detailed sequence of operations. These dispatching rules could be implemented off-line at the scheduling level where the trial solutions are tested against a simulator. When an acceptable sequence of operations is determined, this sequence is passed to factory control as a set of (primal) targets.

Alternately, we could implement the dispatching rules at the factory control level in the form of a real-time control algorithm which chooses the next operation to run on each machine. In this configuration, the scheduling level would generate the priority economics for each operation which would in turn influence the dispatching logic at the control level.

Costs and Economics - A Recap

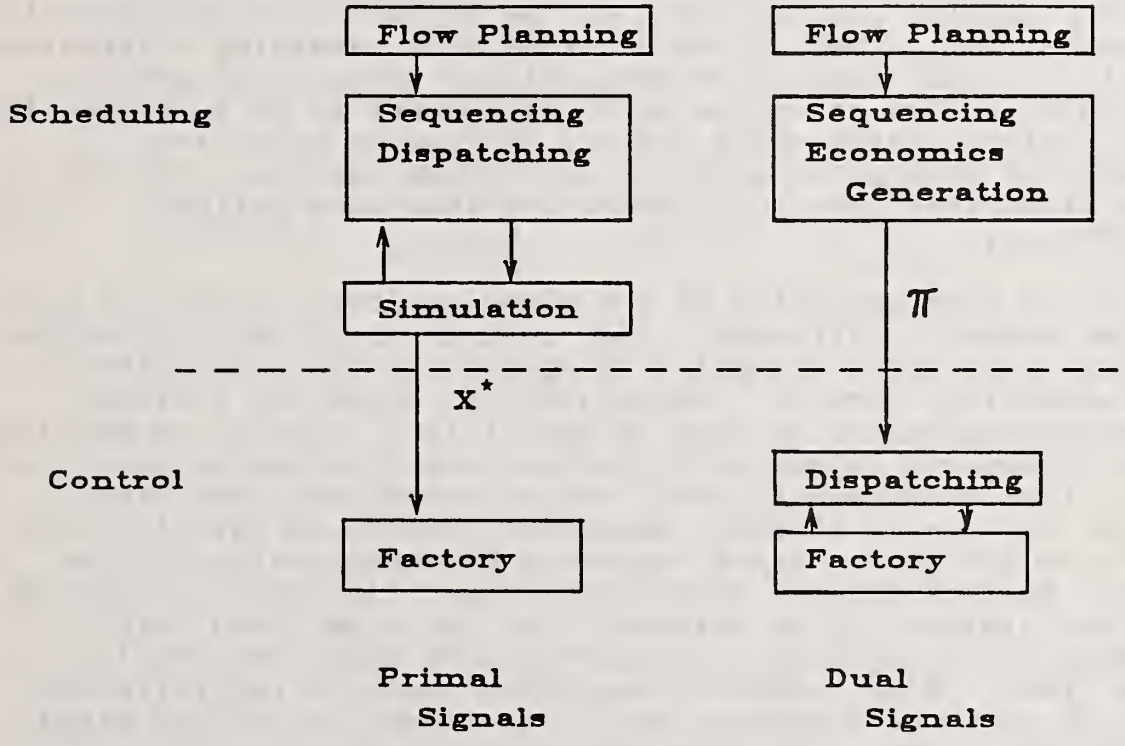
We have alluded to costs and economics several times in this paper. Our primary discussion has dealt with the choice of an economic base so that costs are passed from level to level within the integration framework consistently. Also, we have discussed how algorithms incorporate penalties reflecting relaxed constraints in their objective functions.

Let us now stand back and examine our hierarchical framework more exclusively from the point of view of cost.

First, the broad cost objective of our structure of algorithms as they are applied in concert at any given level of the hierarchy is to reduce the sum of all marginal costs associated with that level. The components of marginal cost may vary in number and definition from one level to the next. Let us consider an example. Figure 10 reflects the primary components of marginal cost associated with the scheduling level; that is, all real costs that will vary with the schedule for the duration of the scheduling horizon, say four to five days. These include

Figure 9

Scheduling/Control Interface

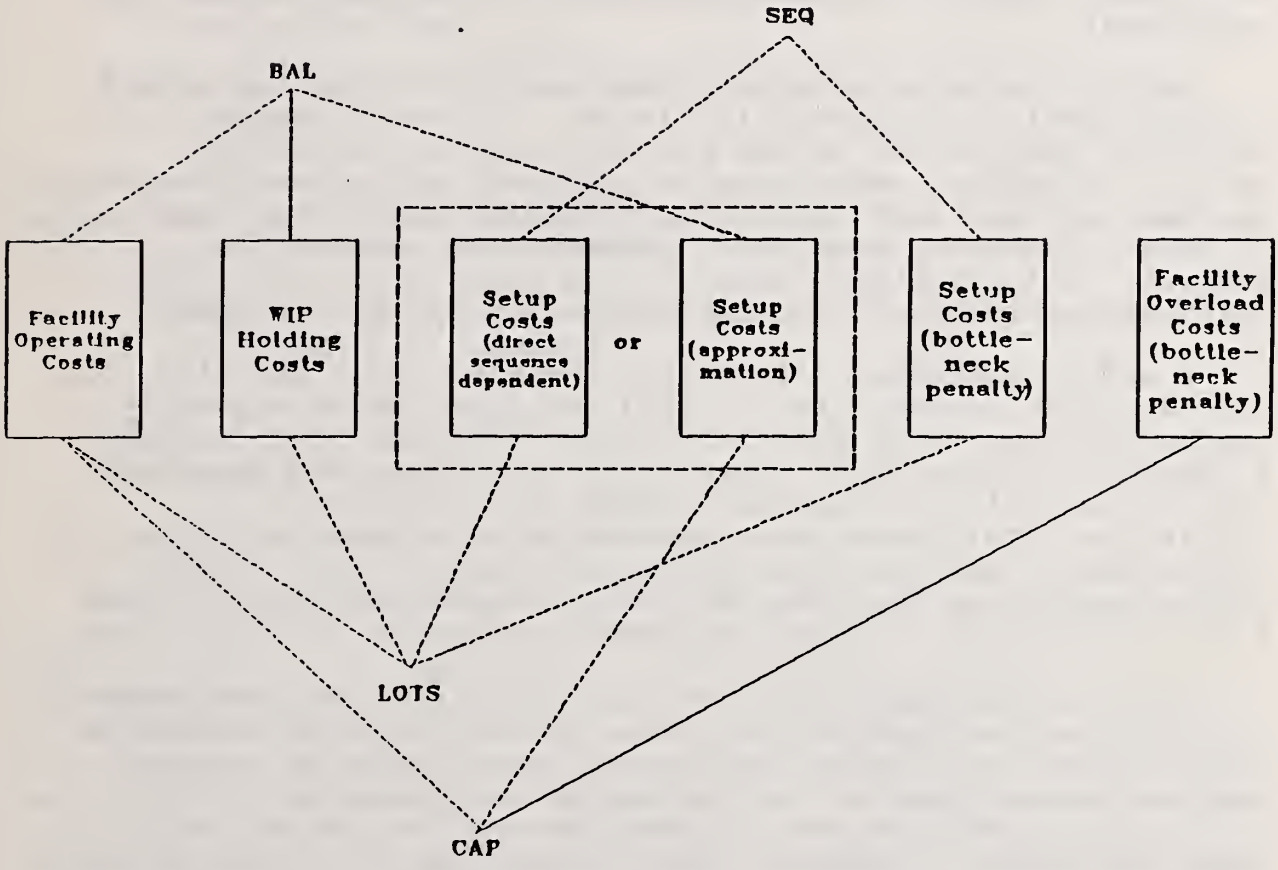


- facility operating costs
- work-in-process holding costs
- direct sequence dependent setup costs
- setup-driven bottleneck penalty costs
- facility overload-driven bottleneck penalty costs

Our approach in this environment to finding a good schedule is first to develop a greedy JIT schedule (see above) and then use various algorithms as appropriate to the human scheduler's specific objectives to improve the schedule. The algorithms work within a framework of total marginal cost but each individual algorithm may incorporate only a subset of these costs as its objective to minimize, given its limited function. The components of marginal cost used by the key algorithms appear in Figure 10. BAL attempts to adjust facility preferences so that expensive setups tend to be reduced. In doing so, it trades off facility operating cost and WIP holding costs against a time period based approximation of setup costs. SEQ on the other hand is examining alternative detailed sequencings of process batches within time periods. Accordingly, its objective is to resequence so as to reduce the sum of direct setup costs and any bottleneck penalties associated with setup over the scheduling horizon. The CAP and LOTS algorithms have their objective functions defined accordingly.

If we were operating at the planning level, Figure 10 would appear somewhat different. Time periods would vary in length perhaps from weeks to months as opposed to hours or shifts in the scheduling horizon. Our objectives would not include detailed sequencing of jobs on facilities. Rather, we would be more interested in making decisions regarding how to best load facilities from week to week, how to order long lead time parts, and how to schedule aggregate production lots from one month to the next. Hence, sequence dependent setup related costs, falling largely within our longer time periods would be neither visible nor of interest. On the other hand, WIP holding costs may play a relatively more major role during this time frame. Also, marginal operating costs of facilities may include costs of fixtures, etc., which were considered fixed within the three to four day scheduling horizon and hence not considered in the scheduling decision. The user's choice of algorithms would also change. SEQ would not be used in this time frame. However, CAP and LOTS would be used but with revised definitions of the marginal cost components of their objectives. The computational structure of each algorithm would remain fixed regardless of the level of the hierarchy within which is was applied.

Figure 10


Costs Considered at the Scheduling Level
SEQ
BAL
Facility Operating Costs
WIP Holding Costs
Setup Costs (direct sequence dependent)
or
Setup Costs (approxi-mation)
Setup Costs (bottle-neck penalty)
Facility Overload Costs (bottle-neck penalty)
LOTS
CAP

Implementation Difficulties

As evidenced by some of the references given above, many of the ideas and algorithms set forth in this paper as as old as the hills. Why haven't they been applied more in practice? In the past, data bases weren't developed to the extent where manufacturing data was under control. In addition, computational power wasn't available in the manufacturing environment where it was needed. Today, both of these restrictions have been lifted; manufacturing data is being captured and organized and computational power is becoming available in almost any form imaginable. However, there do remain some stumbling blocks to the successful implementation of a P/S/C hierarchy--some technical, some organizational and behavioral.

One of the main technical stumbling blocks centers around the aggregation/disaggregation issue. Different levels (and different algorithms) in the hierarchy contain different amounts of detail. When passing information from one level to another, we must deal with the aggregation and disaggregation of data. There has been some academic work done on the subject. Unfortunately, most of this work is not in a form which is applicable to the implementation of our hierarchy.

A second technical difficulty is making the transition from discrete time (production periods) in higher level models to continuous time (date and time of day) in lower level models. In making the transition, inventory projections and capacity utilization calculations must change from discrete to continuous. This transition problem is especially difficult in the automated manufacturing environment where there is more pressure to bring the planning and scheduling functions closer to (continuous time) real-time factory control.

On the organizational side, one of the main implementation difficulties is the lack of manufacturing (marginal) economics information available at the factory level. One of our main implementation tasks is helping operations personnel to estimate marginal cost data appropriate to the given P/S/C level since most traditional cost accounting data is of little value in computing marginal cost trade-offs.

The P/S/C Hierarchy and Economies of Scope

Research into the economic benefits of automated manufacturing has surfaced a term, economies of scope, to describe these benefits. Economies of scope are most easily defined and described in comparison to the more familiar notion of economies of scale. Jelinek and Goldhar (1986) characterize economies of scale thinking in terms of well known "rules":

- o production costs decrease as factory size increases
- o inventory is to function as a buffer to protect against input disruptions
- o product variation should be minimized in favor of long production runs
- o a large proportion of production costs are separable and variable

They describe economies of scope in contrast as typically longer term and frequently difficult to quantify benefits that accrue from automated manufacturing due to such attributes as:

- o improved production speed and flexibility in responding to changes in demand mix
- o greater product variation
- o reduced WIP inventories
- o reduced changeover (setup) times and costs

They suggest that these attributes can lead to strategic benefits over the long term such as increased market share or even survival in a given market. For example, a firm implements automated manufacturing to improve product quality or to improve response time in providing replacement parts, or to be able to be competitive in smaller market segments that it was previously prohibited from entering. The economies of scope logic emphasizes that the benefits are typically neither direct nor immediate but strategic nonetheless. Furthermore, currently popular investment decision making frameworks such as return on investment (ROI), payback, and net present value (NPV) have difficulty capturing economies of scope due to their emphasis on the near term.

Our hierarchical framework incorporates a structure that works to ensure that the economies of scope reflected by the automated manufacturing environment are, indeed, realized:

o a function wide data base framework allows data that is needed at all P/S/C levels to be entered with a minimum of duplication

o an algorithmic structure allows primal and dual information to be passed between levels to ensure that decisions are internally consistent, e.g., long lead time parts ordered at the flow planning level are used economically at the scheduling level

o WIP inventories are kept to a minimum so that they will not become stranded (resulting in waste) when the mix of demands on the factory changes suddenly

o costs of setups are balanced carefully with facility operating costs, WIP holding costs, and bottleneck effects on overall factory throughput in an effort to reduce the sum of all marginal costs associated with the automated manufacturing environment in the context of the given time horizon.

Summary

This paper describes an integration framework in which the Planning/Scheduling/Control (P/S/C) hierarchy can be imbedded in the context of the automated manufacturing environment. This framework consists of a data base structure which spans the P/S/C functions. It includes a collection of algorithms, each with a focused objective, which operate on the data base and are selectively applied in concert by the user.

The integration framework combines several well known concepts and computational approaches in new ways. It is structured to allow the use of a consistent economic or cost base across all levels. It uses primal and dual decomposition techniques to transfer information both between levels of the P/S/C hierarchy and within a given level. The specialized algorithms incorporate Lagrangean relaxation techniques to cost out the impact of not meeting the various constraints of the classical planning/scheduling problem.

The paper traces the development of the key algorithmic concepts over the last twenty years that provide the building blocks for the framework. Many of these methods were first addressed and developed to deal with planning, scheduling, and control problems in the process industry. As discrete manufacturing becomes more flexible and automated, these process industry solutions become more and more applicable to the discrete manufacturing environment.

Problems encountered in implementing the framework fall into two classes: technical and organizational. One of the key technical problems is dealing with how to aggregate and disaggregate data efficiently. Another is how to design algorithms that work together smoothly when making the transition from discrete time to continuous time modeling formulations. A primary organizational problem is estimating the various marginal cost data elements that can be utilized by the framework in factory environments where most cost data is maintained in traditional accounting systems and is not marginal in nature.

A primary thrust of this paper has been to show that some well-chosen traditional operations research techniques take on new life when implemented on a function-wide data base.

References

Graves, S. C. 1982. Using Lagrangean Techniques to Solve Hierarchical Production Planning Problems. Management Science 28, 3, 260-275.

Jelinek, M. and J. Goldhar. 1986. Economics in the Factory of the Future. CIM Review Winter 1986, 21-28.

Hax, A. and H. Meal. 1975. Hierarchical Integration of Production Planning and Scheduling. In Studies in Management Sciences: Logistics, e. M. Geisler, North-Holland-American Elsevier, New York.

Jaikumar, R. 1974. An Operational Optimization Procedure for Production Scheduling. Computers and Operations Research Vol 22, No. 2, 191-200.

Lin, S. 1965. Computer Solution of the Traveling Salesman Problem. Bell System Tech. Journal 44, 2245-2269.

Lefkowitz, I. 1977. Integrated Control of Industrial Systems. Phil. Trans. R. Soc. Lond. A. 287, 443-465.

Maxwell, W., J.A. Muckstadt, et.al. 1983. A Modeling Framework for Planning and Control of Production in Discrete Parts Manufacturing and Assembly Systems. Interfaces 13, 6, 92-104.

Mesarovic, M. D., D. Macko and Y. Takahara. (1970). Theory of Hierarchical Multilevel Systems. Academic Press, New York.

Odrey, N., Nagel, R. 1986. Critical Issues in Integrating Factory Automation Systems. CIM Review Winter 1986, 29-37.

Tyagi, M. 1968. A Practical Method for the Truck Dispatching Problem. J. of the Operations Research Society of Japan 10, 76-92.

Wagner, H. M. and T. Whitin. 1958. Dynamic Version of the Economic Lot Size Model. Management Science 5, 1, 89-96.

INTELLIGENT MANUFACTURING PLANNING SYSTEMS

David Liu
Electro-Optical and Data Systems Group
Hughes Aircraft Company
U. S. A.

ABSTRACT

Most automation endeavors in manufacturing require at least some manual direction. Now, Artificial Intelligence (AI) systems which embody facts, logic, and production rules can mimic the reasoning of manufacturing experts. This paper describes an AI system which creates manufacturing planning automatically from CAD databases.

Introduction

Manufacturing planning determines the overall sequence of operations, prescribes the step-by-step instructions of each operation, and schedules the operations. Many firms have developed computer-aided process planning (CAPP) systems to reduce the time required to sequence the operations. These systems can produce skeletal process plans. However, they still require a great deal of manual direction and input.

Today, some manufacturers are attempting to completely automate manufacturing planning. Using expert systems, an application of Artificial Intelligence, these firms are developing software that mimics the reasoning of manufacturing planners. Along with information from the engineering database, embodied facts, logic, and rules are applied to generate manufacturing plans automatically.

AUTOMATED PROCESS PLANNING SYSTEM

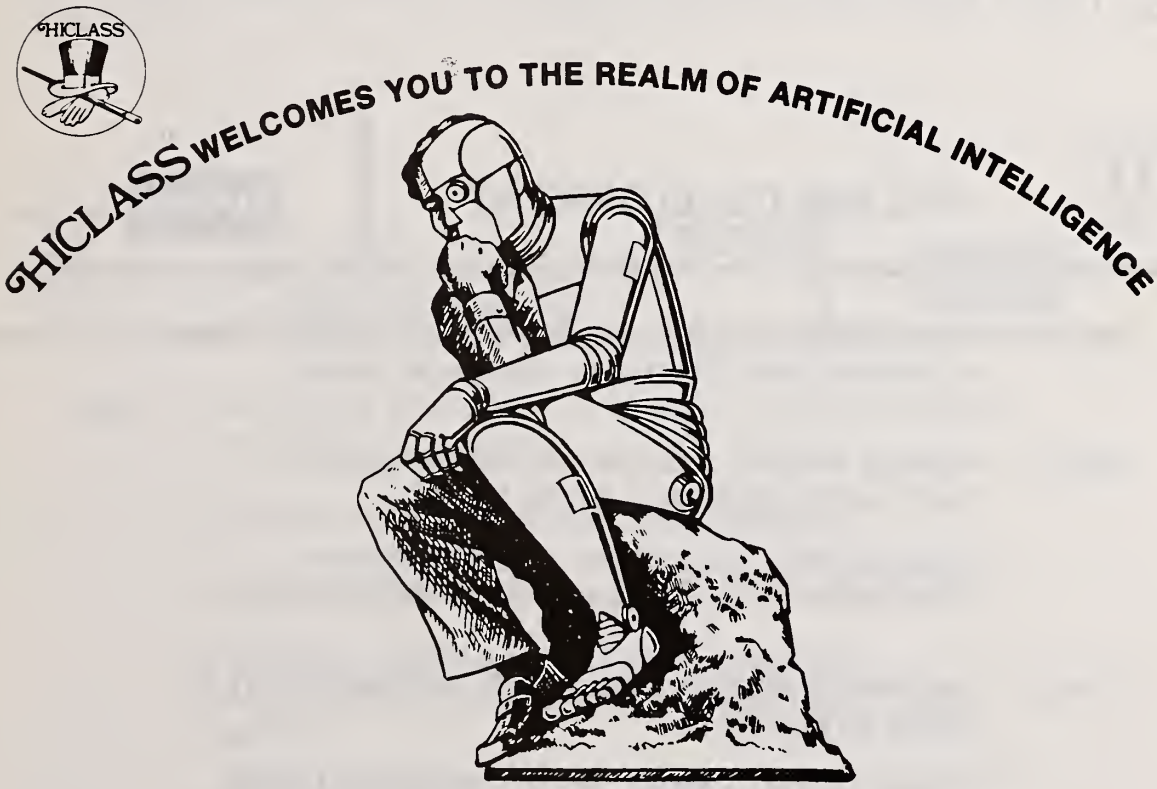
HUGHES

REQUIREMENTS

1. **GENERATIVE**
2. **FLEXIBLE, PORTABLE**
3. **HUMANLIKE INTELLIGENCE**
4. **EASY TO USE**
5. **FREE PEOPLE FOR JOBS COMPUTER CANNOT HANDLE**
6. **INTEGRATE WITH REST OF MANUFACTURING SYSTEM**
7. **AVAILABLE EXPERTS**
8. **STABLE TASK DEFINITION**

One of the first companies to implement such a system is Hughes Aircraft Company's Electro-Optical and Data Systems Group. The firm has developed the Hughes Integrated Classification Software System (HICLASS)™ to integrate design and manufacturing. The objective is to electronically deliver computer generated planning throughout production. This enhances product quality, reduces support costs, increases flexibility and yields greater responsiveness.

To generate this on-line manufacturing planning, Hughes determined that the system must be capable of capturing manufacturing knowledge. The manufacturing knowledge is used to deduce required manufacturing operations from engineering databases. The design defines the problem, and available data includes graphical representation, engineering notes, design and manufacturing specifications, special manufacturing instructions, and inspection criteria.



Evolution

Hughes made several attempts at developing such a system before settling on the current expert system. In 1982, the work began with a generative manufacturing planning system in which design features were matched to specific manufacturing operations. This system used so-called binary decision trees to produce process plans for printed circuit board (PCB) fabrication. Process networks and flows describing manufacturing operations were set up in this decision tree format. Although successful, the trees were too difficult to build. Moreover, process plannninging resembles an unstructured, 3D network, so mapping this network onto binary decision trees left out essential information and led to a loss of flexibility.

A second generation version was then built using multi-branch decision trees. Such trees were easier to build and modify. This system was benchmarked for three applications, including PCB fabrication, PCB assembly, and mechanical part fabrication. PCB applications proved that multi-branching worked well. Mechanical part applications, however, revealed the shortcomings of manually stepping through the decision trees. Planners would lose perspective of the problem after traversing a few levels into the decision trees.

HICLASS DEVELOPMENT HISTORY

HUGHES

1982 — PROTOTYPE PRINTED WIRING BOARD FABRICATION PLANNING SYSTEM USING DECISION TREES

1983 — PASCAL EXPERT SYSTEM PROGRAM WHICH CAPTURED LOGIC IN IF/THEN RULES

— DEMONSTRATION EXPERT SYSTEM WHICH CREATED PROCESS PLANS FOR 2 ASSEMBLIES

1984 — SYSTEM IN PRODUCTION FOR 2 ASSEMBLIES AT 5 WORK STATIONS

— BEGAN DEVELOPMENT OF "C" PROGRAM USING MORE COMPLEX IF/THEN/ELSE RULES AND ENCHANCED CONCEPTUAL STRUCTURES

This system, called the Hughes Integrated Classification Software System (HICLASS)™, used group technology software to classify parts and generate process plans. As an enhancement, software was included to infer part features based on part geometry. These inferred-part features guided the system in traversing decision trees and picking out essential manufacturing steps.

Inferring part features, however, proved to be a complex task. This early system required a significant amount of software and time to determine a route through decision trees. Such a structure was still not an optimal representation of manufacturing planning.

An entirely new system, also called HICLASS™ Software System, taking the knowledge-based, expert systems approach was then developed. The system's capability to automatically extract part features from CAD frees planners from describing the part. This software deduces required manufacturing operations, instructions, and equipment programs from the engineering database through heuristics that represent manufacturing experience and intuition.

HICLASS CURRENT DEVELOPMENT

HUGHES

1985 — TRANSFER ASSEMBLY PLANNING EXPERT SYSTEM TO NEW RULE FORMAT

— CREATE PLANNING FOR 32 ASSEMBLIES

— SYSTEM IN PRODUCTION AT 60 WORK STATIONS

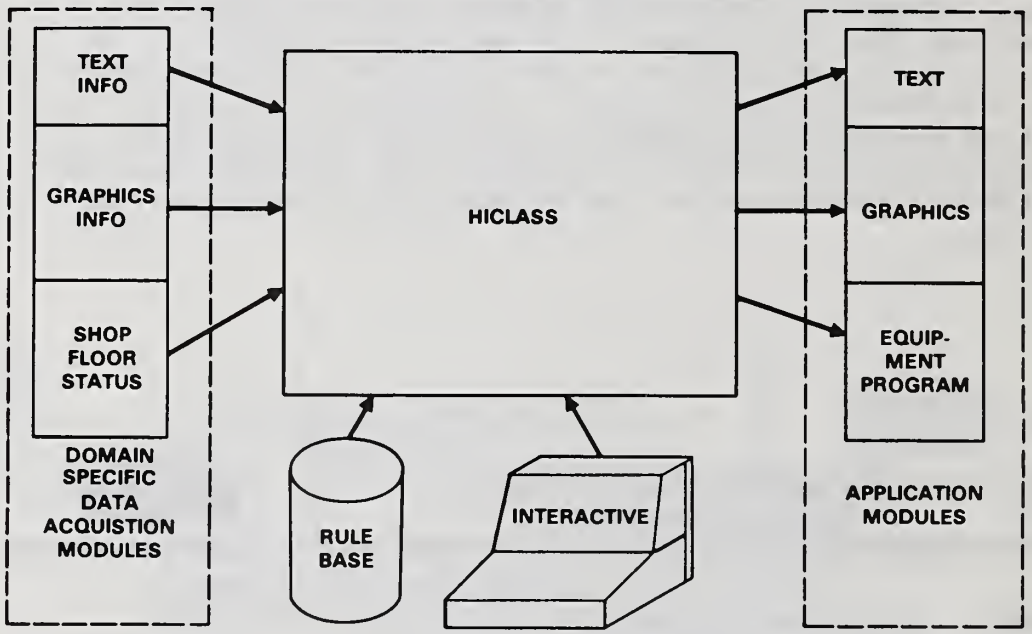
— DEVELOP A GENERAL NATURAL LANGUAGE PARSER TO INTERPRET ENGINEERING DRAWING NOTES

— BEGIN DEVELOPMENT OF PROTOTYPE PRODUCIBILITY ANALYSIS SYSTEM FOR PRISMATIC PARTS

The software operates in three phases to accomplish automated manufacturing planning. First, an interpretation phase defines the problem by translating engineering databases into "tokens" that can be processed as symbols rather than numbers. Next, a reasoning phase solves the planning problem by evaluating constraints and goals. Here, the symbolic processing continues until a conclusion or conclusions can be drawn. Finally, a presentation phase translates computer graphics which are sent to terminals on the shop floor. In the future, the presentation phase will also translate results into equipment programs that control the machinery.

HICLASS SYSTEM FUNCTIONAL DESCRIPTION

HUGHES



CURRENT CONFIGURATION

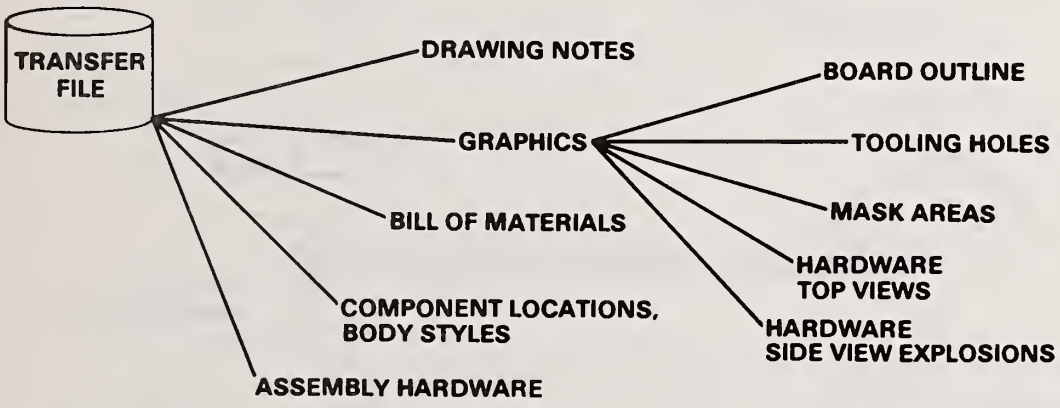
COMPUTER SYSTEM	APOLLO ENGINEERING WORK STATIONS
TERMINALS (5)	TEKTRONIX, COLOR
OPERATING SYSTEM	UNIX
LANGUAGE	PASCAL, C
NETWORK	APOLLO DOMAIN
INTERFACE	HP3000, CV/CALMA, IBM

Interpretation

Design engineers use interactive graphics to capture product descriptions into the engineering database, including geometric models, hidden-part attributes, and bills-of material. Also included are engineering notes, which contain part specifications, special instructions, and general design comments.

TRANSFER OF CAD DATA TO MANUFACTURING

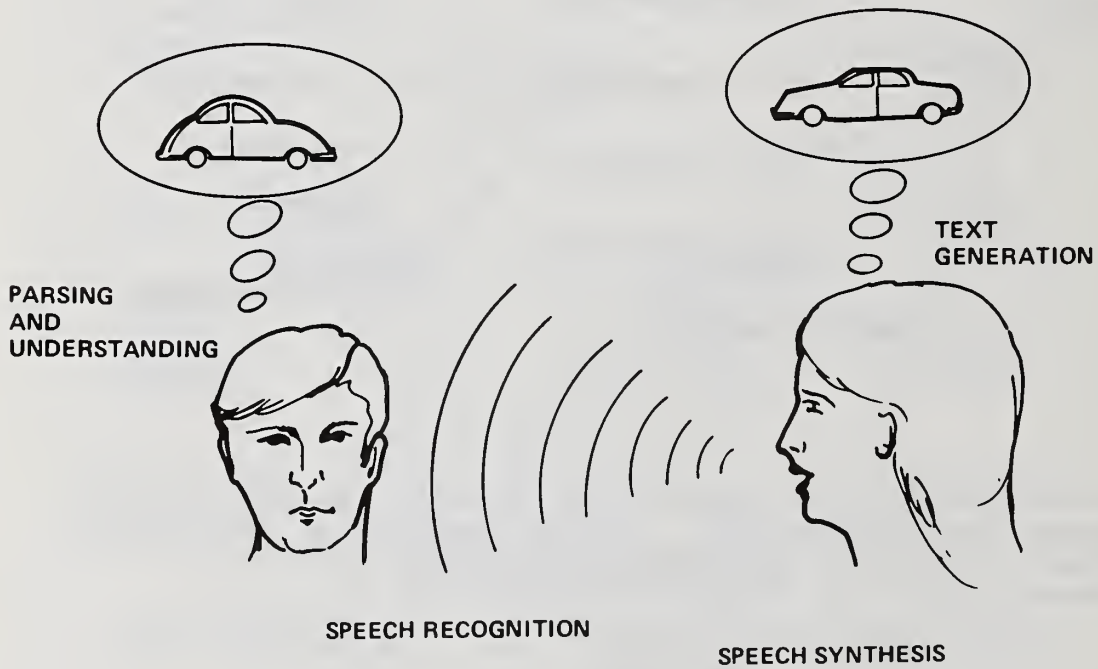
HUGHES



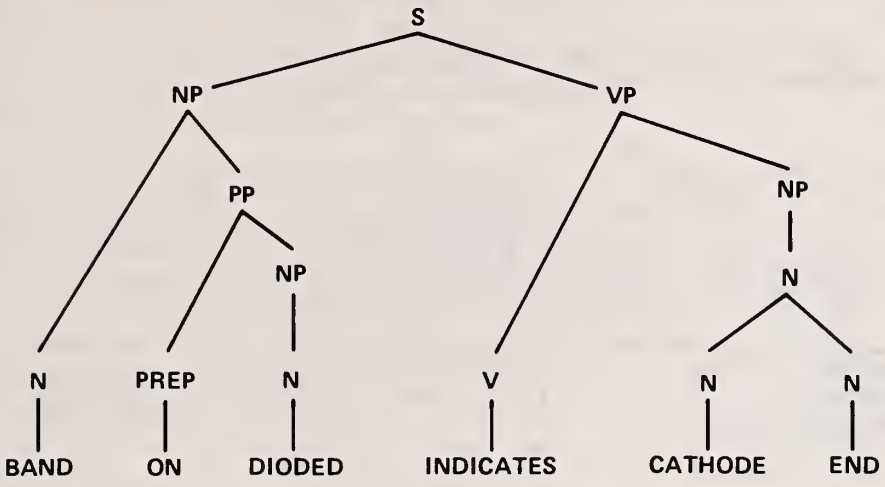
A complete product description is essential for correct manufacturing planning. If, for example, a "heat treat" note is missing, the resulting process plan may be incomplete.

Another facet of Artificial Intelligence known as natural language processing helps interpret engineering notes for use in manufacturing planning. The system uses a mixture of syntax parsing and semantics interpretation methods to understand these notes. Syntax provides grammar and semantics provides meaning. Both are represented by networks. Nodes in these networks represent word categories such as verb or noun, while arcs denote relationships between node (word or phrase) classes. Semantic interpretation provides meaning to words and phrases extracted from the engineering notes.

NATURAL LANGUAGE PROCESSING



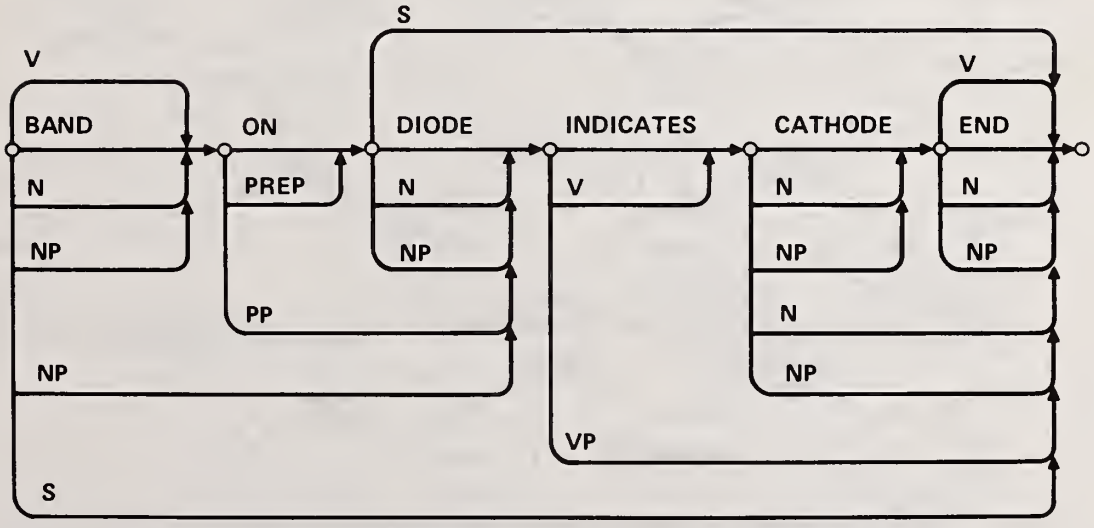
PHRASE STRUCTURE GRAMMER



S → NP VP
NP → N PP
NP → N
N → N N
PP → PREP NP
VP → V NP

N → BAND
N → DIODE
N → CATHODE
N → END
PREP → ON
V → INDICATES

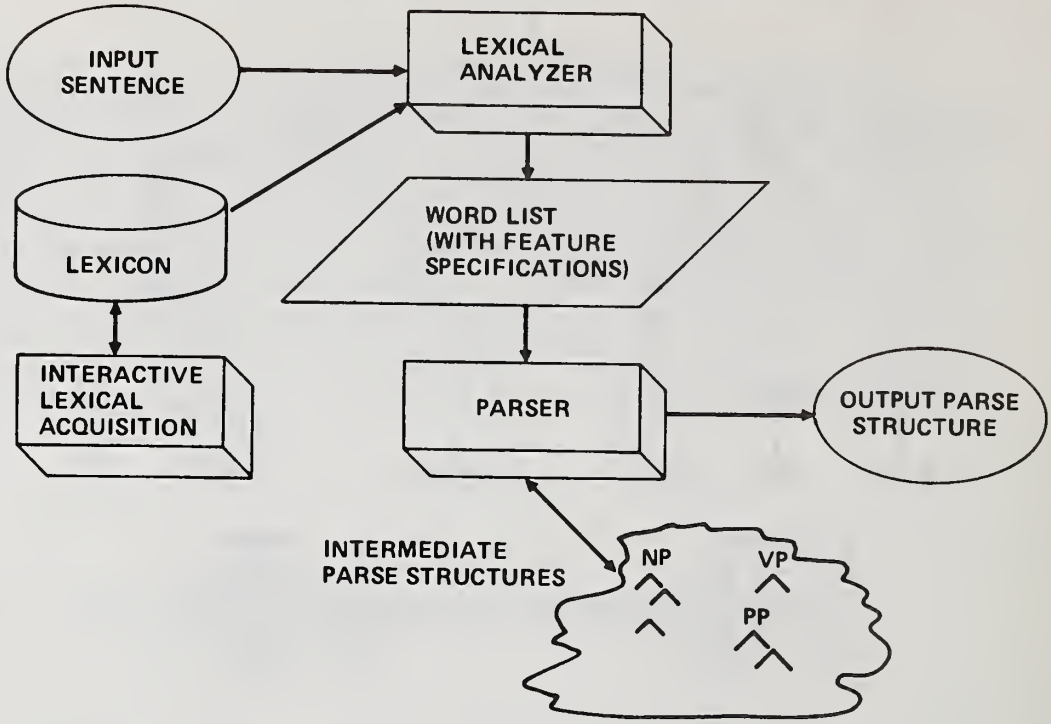
BOTTOM-UP PARSE STRATEGY



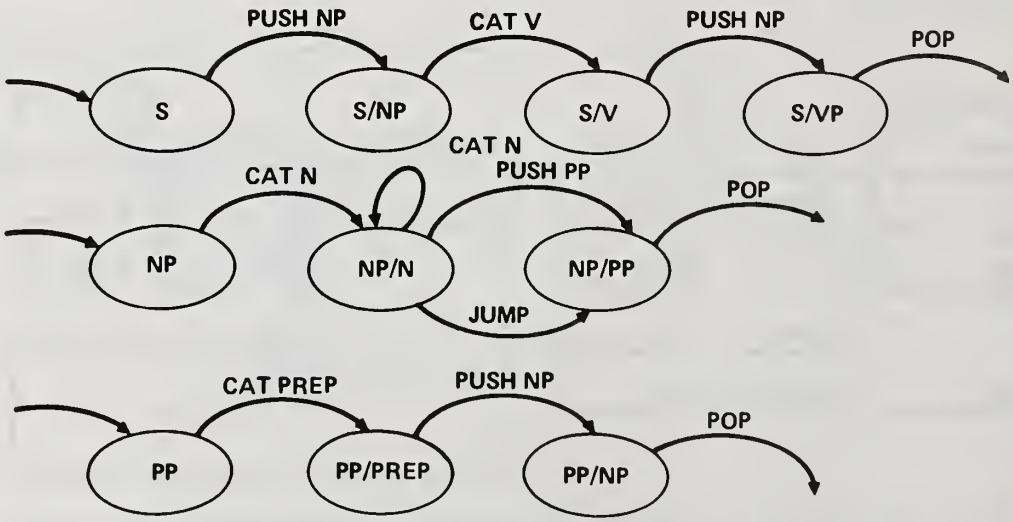
S → NP VP
NP → N PP
NP → N
N → N N
PP → PREP NP
VP → V NP

N → BAND
N → DIODE
N → CATHODE
N → END
PREP → ON
V → INDICATES

HICLASS NATURAL LANGUAGE SUBSYSTEM



AUGMENTED TRANSITION NETWORKS



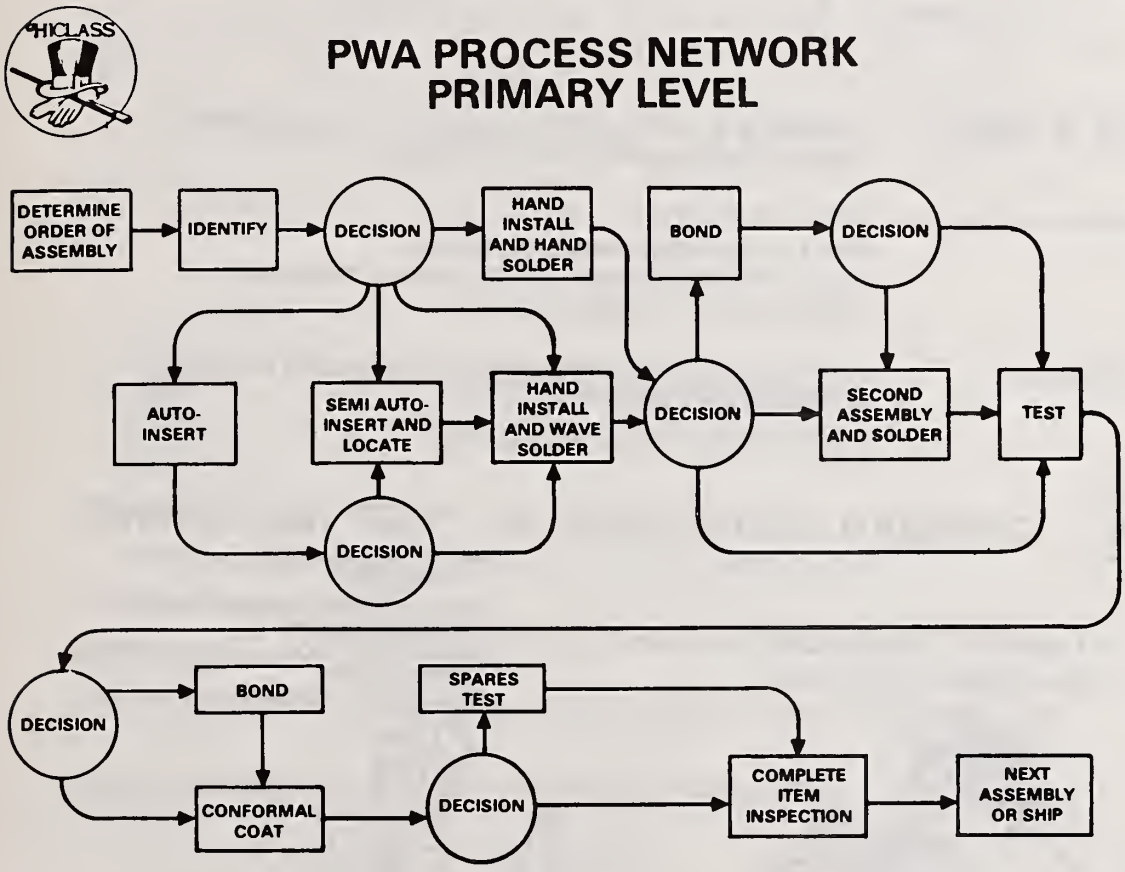
INPUT SENTENCE: BAND ON DIODE INDICATES CATHODE END

	STATE	ARC	CURRENT WORD		STATE	ARC	CURRENT WORD		STATE	ARC	CURRENT WORD
								15.	NP	CAT N	CATHODE
1.	S	PUSH NP	BAND	7.	NP	CAT N	DIODE	16.	NP/N	CAT N	END
2.	NP	CAT N	BAND	8.	NP/N	CAT N	INDICATES	17.	NP/N	CAT N	
3.	NP/N	CAT N	ON	9.,	10., 11.			18.,	19., 20,		
4.	NP/N	PUSH PP	ON	12.	NP/PP	POP	INDICATES	21.	NP/PP	POP	
5.	PP	CAT PREP	ON	13.	S/NP	CAT V	INDICATES	22.	S/VP	POP	
6.	PP/PREP	PUSH NP	DIODE	14.	S/V	PUSH NP	CATHODE				

Reasoning

Manufacturing knowledge is represented by networks and production rules. These networks and production rules provide the domain specific knowledge required to generate manufacturing plans automatically from the engineering database.

Networks describe the possible flow of parts through manufacturing and are composed of nodes which represent actions (operations) or decisions (movements). Arcs denote an association between two nodes. Networks can also depict the importance of precedent and sequel relationships. For example, a node placed in front of another implies a sequence of events.

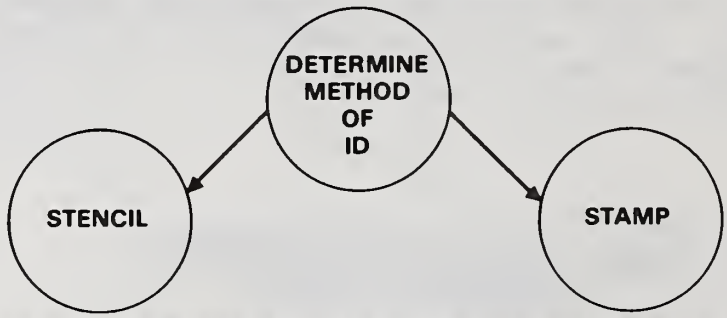


PERT diagrams are a common example of a network and have been used for many years to analyze factory flow. Unlike PERT diagrams, these decision networks capture meta-knowledge to guide selection and evaluation of rules. The flow and control of a planner's reasoning can be modeled by such networks.

Networks can also be nested; that is, one network can be embedded within another. These structures provide multiple levels of abstration to a problem which relate to different levels of detail. At higher levels, for example, a network can represent the flow of parts through an entire factory. At lower levels, a network can represent a flow within a work center.

HOW DO WE MAKE THIS DECISION?

HUGHES



RULE OF THUMB: IF THERE IS A NOTE ON THE DRAWING "STENCIL BOARD . . . " THEN THE BOARD MUST BE STENCILLED

DEEPER QUESTIONS:
- WHAT IS A NOTE?
- WHAT IS AN ENGINEERING DRAWING?
- WHERE ARE THE NOTES LOCATED ON THE DRAWING?
- HOW DOES ONE INTERPRET THEM?

IMPLICATION: JUST LIKE A HUMAN, HICLASS MUST BE TAUGHT IN ORDER TO BECOME AN EXPERT!

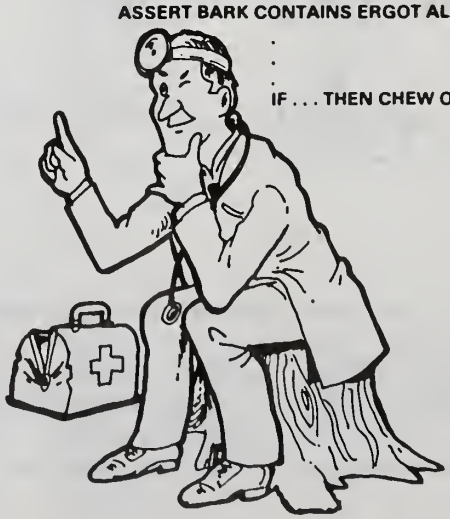
CAVEMAN KNOWLEDGE VS. DEEP REASONING

IF I HAVE A HEADACHE THEN CHEW ON BARK

- ASSERT HEADACHE COULD BE CAUSED BY PRESSURE . . .
- ASSERT PRESSURE COULD BE CAUSED BY VESSELS DILATION
- ASSERT ERGOT ALKALOID CAUSES VESSELS CONSTRICTION
- ASSERT BARK CONTAINS ERGOT ALKALOID

. . .

IF . . . THEN CHEW ON BARK



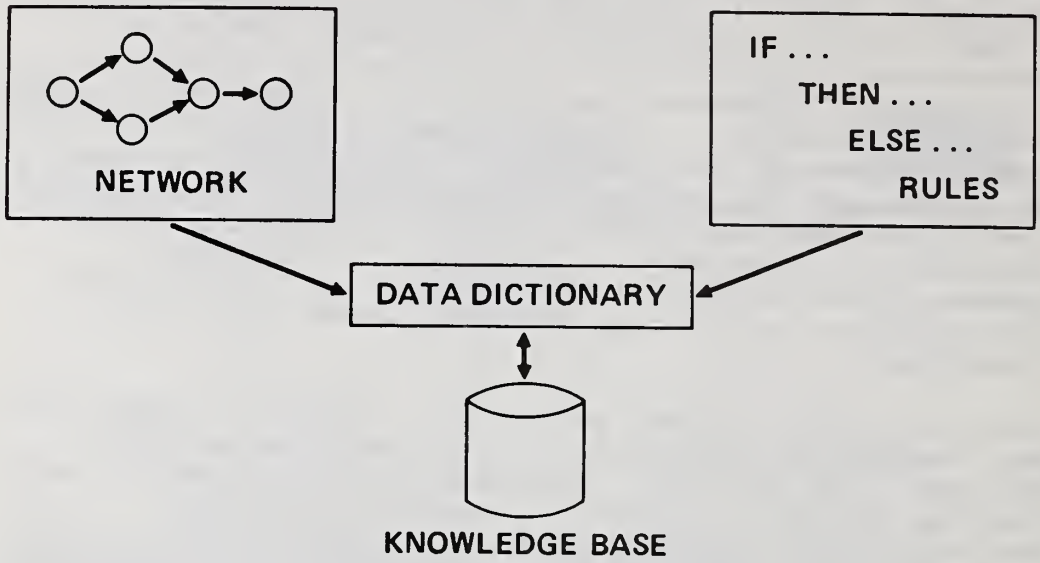
Manufacturing experience and intuition are fed to the inference engine in the form of production rules. IF-THEN logic is an effective and common method of representing this heuristic knowledge. In an expert system the IF statement is called the antecedent, while the THEN statement is the consequent. As with high-level programming statements, both the antecedent and the consequent can contain complex logical expressions. These functions manipulate and evaluate slots, which resemble variable names in high-level languages, however, this IF-THEN logic is evaluated in a declarative fashion, so ordering of the IF-THEN logic is unimportant.

Other functions further permit the software to mimic expert reasoning. Solutions to problems are rarely a straight "yes" or "no". Instead, humans arrive at conclusions based on partial or uncertain evidence. Fuzzy-set logic is designed to deal with problems where there are many shades of gray, such as in manufacturing planning. Production rules contain "certainty" and "threshhold" to facilitate fuzzy-set logic.

HICLASS INFERENCE ENGINE CAPABILITIES

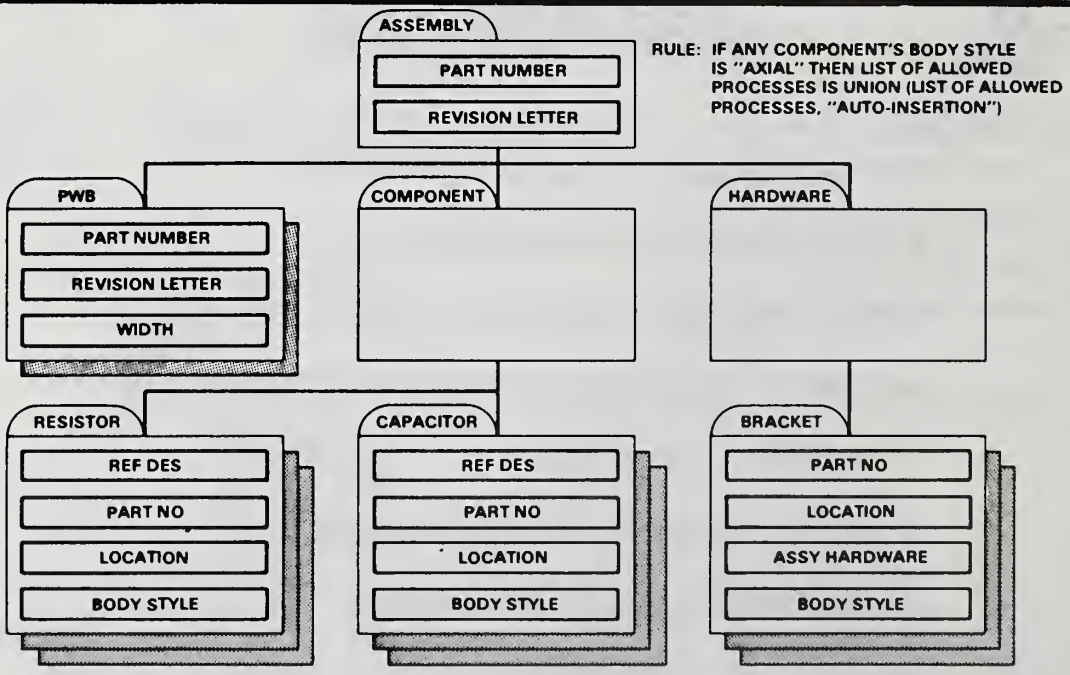
- IF ANTECEDENT THEN CONSEQUENT
- COMPLEX LOGIC EXPRESSION
- FUZZY MODEL (CERTAINTY AND THRESHOLD)
- LIMITED LEARNING (EXPERIENCE TABLE)
- STABILITY ANALYSIS
- RESOLVE CONFLICTS (PRIORITIES)
- MULTIPLE LINES OF REASONING

THE KNOWLEDGE BASE



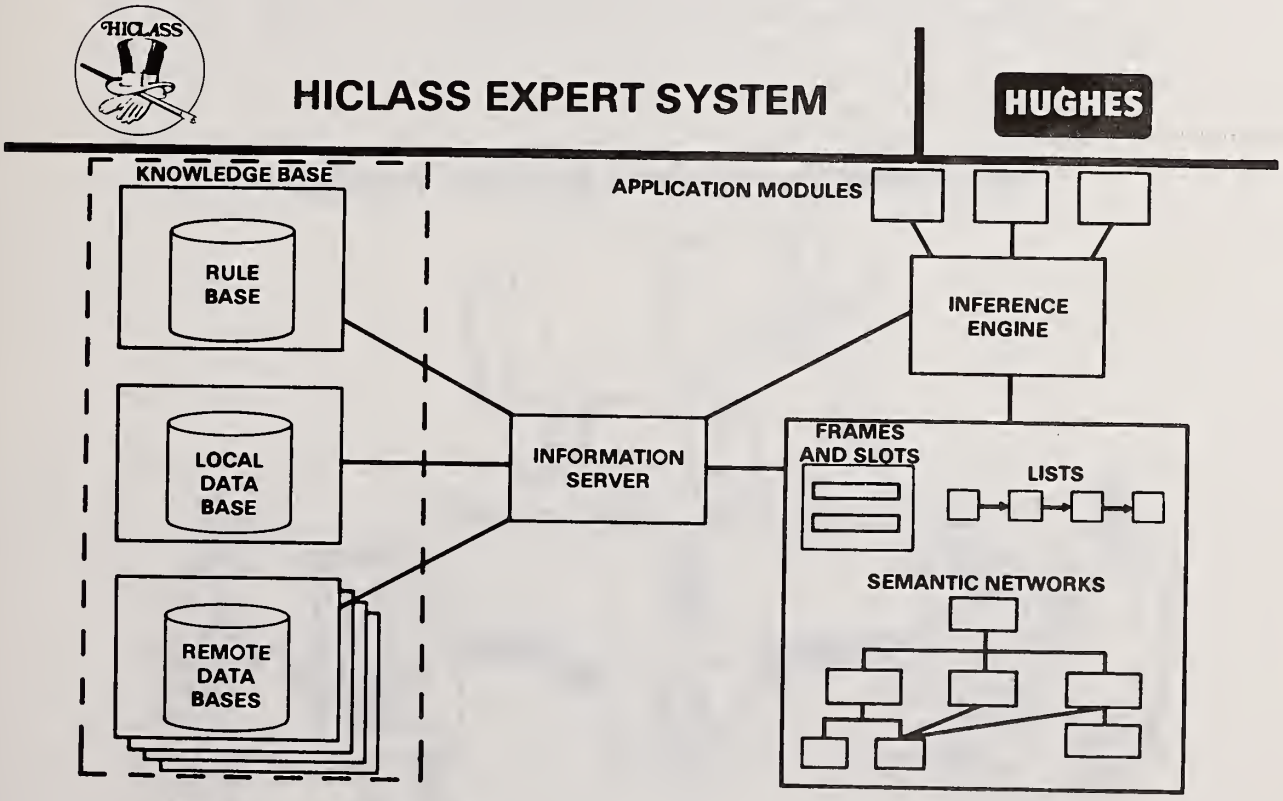
EXAMPLE OF KNOWLEDGE STORAGE STRUCTURE

HUGHES



Another problem the software must handle is non-determinism, which occurs when a single event results in a number of alternatives. For example, when the expert system recognizes that a cylindrical cavity needs to be created in a part, the alternative operations are drill, bore, broach, or ream. The system uses a ranking of priorities to resolve these conflicts. If priority is based on lowest cost alone, the system will choose drilling.

The expert system uses forward chaining (or forward inferencing) to generate manufacturing plans since the task involves creating a strategy for manufacturing a part from start to finish. In PCB assembly, for example, a planner starts with a bare board and works with it until all components have been assembled. Expert systems can also be built with backward inferencing, in which a problem is worked through from finish to start. In such a system, the PCB assembly would regress until it became a bare board. Since planners are unaccustomed to working with backward inferencing, it was not used.

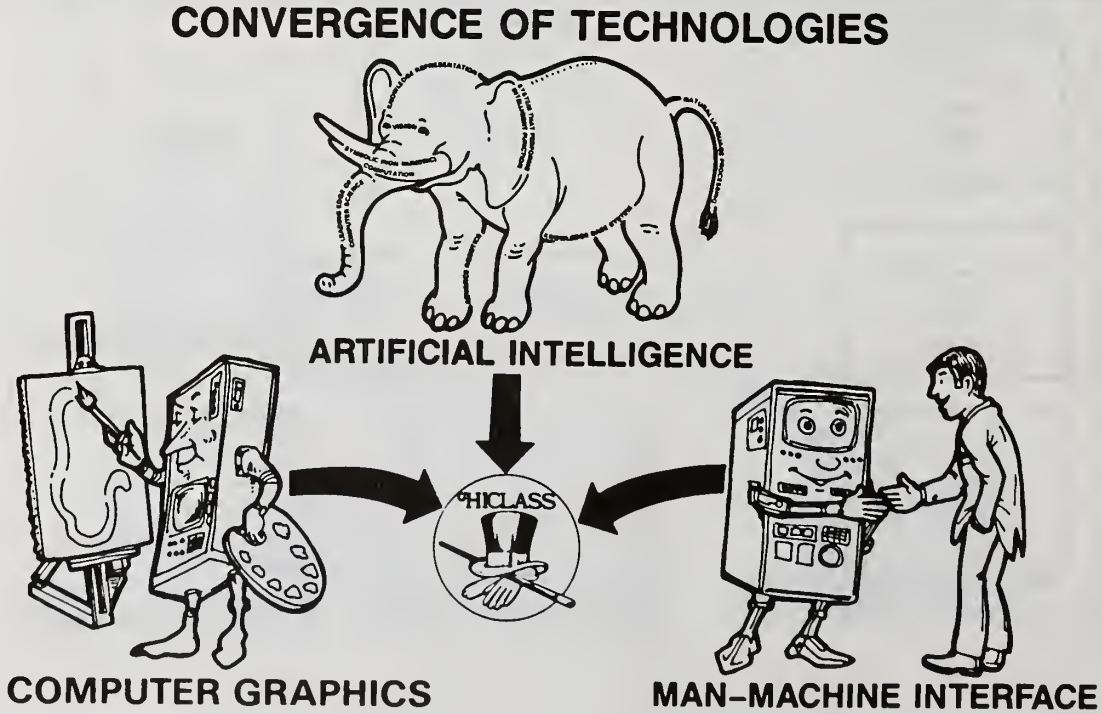


Man-Machine Interface

Automation of manufacturing planning involves computer interfaces on three levels. Experts (manufacturing and design engineers) must "teach" the system their knowledge. End users (workers who assemble the product) receive the finished plans. System implementers (typically computer scientists) develop the software which transforms expert knowledge into the finished plan. Since the computer literacy for each class of user varies, the interfaces much match the level of each.

The interface for experts is designed to acquire knowledge quickly. Typically, manufacturing engineers can best describe knowledge with PERT diagrams (process networks) and English-like statements for production rules. These experts use graphics-oriented engineering workstations to enter process networks into the system. One benefit of the system is that the knowledge representation is explicit and not implicit. In other words, these networks are used directly and do not have to be transformed into other knowledge representations.

Thus, no information is lost since there is no mapping involved. In addition, manufacturing engineers teach the system discrete logic required for manufacturing. This logic is represented by production rules.



Presentation

The end-user interface is designed for teaching and presenting instructions on how to build a product. The product can be built by either humans or robots. For now, we will concentrate on the human aspects. The instructions can be relayed verbally, textually, or graphically. Hughes decided that pictorial representations are such the most efficient method of presenting information, so the expert system generates computer graphics illustrations along with textual instructions for production workers.

In addition to static frames of instructions, the system uses a script to embody information on dynamic rotation, translation, and other commands. Such functions require the host to have high-computational speed because software performs a large number of matrix manipulations to achieve graphical mapping. The hardware utilizes floating-point features and the operating system utilizes large virtual memory to enhance the functionality of the system.

High throughput is also important. The CRT at the factory workstation must display an image as fast as the host receives commands from the production operator. To accomplish this, these terminals must have graphics routines in both hardware and firmware. Since high computational speed and throughput are conflicting requirements in most computers, the systems uses a computer with powerful CPU's coupled to direct memory access (DMA) devices to meet the performance requirements.

Computer Graphics

Graphics are drawn on interactive systems amd are passed to the expert system in either a vector format or through the Initial Graphics Exchange Standard (IGES). For presentation, the system translates results from the decision-making process and combines then with results of the Rule-Driven Graphics Generator module to create assembly scenarios. The System now uses a Plot-10 graphics protocol to send these images to factory-floor terminals. In the future, the system may migrate to some video protocol such as the North American Presentation Level Protocol Syntax (NAPLPS) as well as video disk technology.

Conclusion

Expert systems are an excellent tool for automation. They embody human expertise to perform complex tasks such as manufacturing planning. The expert system holds great potential in the endeavor to automate manufacturing support services (white collar functions).

A REAL EXPERT SYSTEM

HUGHES

DISADVANTAGES

- REQUIRES LOTS OF EFFORT TO TEACH THE SYSTEM THE BASICS

ADVANTAGES

- HICLASS WILL NOT QUIT AFTER 2 YEARS OR RETIRE
- UNIFORM APPLICATION OF EXPERTISE
- CONSISTENTLY ACCURATE PERFORMANCE
- ACCUMULATES KNOWLEDGE FROM MANY EXPERTS
- INTEGRATES KNOWLEDGE FROM MANY DISCIPLINES

POTENTIAL AI APPLICATIONS

HUGHES

ENGINEERING
- CHOOSING SPECIFICATIONS, MATERIALS, PROCESSES
- DESIGN ANALYSIS
- PRODUCIBILITY ANALYSIS

PRODUCTION CONTROL
- PRODUCTION PLANNING

MANUFACTURING ENGINEERING
- CREATION OF PLANNING
- FINAL SELECTION OF MATERIALS, PROCESSES
- DESIGN OF TOOLING FIXTURES
- INTERPRETATION OF DESIGN PARAMETERS

ASSEMBLY AND FABRICATION
- SCHEDULING
- ALTERNATE ROUTING
- FAULT DIAGNOSIS
- ADAPTIVE CONTROL
- EQUIPMENT PROGRAMMING

QUALITY
- INSPECTION INSTRUCTIONS

A DECISION MAKING FRAMEWORK FOR MANUFACTURING SYSTEMS*

by

Prof. George Chryssolouris
Laboratory for Manufacturing and Productivity
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

INTRODUCTION

Modern industrial technology is often faced with the task of producing a wide variety of parts in small to medium batch sizes at the highest possible efficiency. Much effort has gone into designing hardware such as machine tools and material handling devices, as well as equipment layout, in order to meet the objectives of flexible manufacturing. Indeed, with today's available hardware, systems adaptable to a wide range of production requirements can be configured. In the case of Flexible Manufacturing Systems, the control of the system is provided by computer software adequate for operation without human intervention; however, flexible manufacturing can also be carried out by a combination of numerically controlled machines, automated handling devices, and human operators, with the system being controlled by a foreman or a plant manager. Whether fully automated or not, the level of efficiency achieved from such capital intensive systems depends on the decision making procedures for assigning

* The work described in this paper has been supported by CAM-i's Factory Management Program and the paper has been originally presented during the 1985 CAM-i Annual Meeting in San Diego, California.

the system resources to the various production tasks. Two factors are important in considering adequate decision making procedures:

- Flexibility, a primary aim of the system, can be entirely lost if the decision making techniques for resource assignment, whether implemented by man or machine, are either so simplistic that they inadequately account for changes in the manufacturing environment or so complex that the wealth of detail and computational resources they require preclude a sufficiently rapid response to changes in the environment. Traditional scheduling techniques in flexible manufacturing often involve either extensive simulation procedures or lengthy iterative processes. Both approaches seek to bring the implications of ideal models into conformity with the reality of the production environment.

- Decisions reached by most scheduling techniques in use today are optimized with respect to a single criterion. Such an approach requires that all aspects of a system be reduced to that single criterion in a predetermined manner. However, this methodology assumes that the manufacturing environment is static when in fact priorities, requirements, and conditions are constantly changing.

Therefore, decision making techniques which are more implementable yet, at the same time, more responsive to changes must be developed.

This paper describes a method for MAnufacturing DEcision MAking (MADEMA) which attempts to assign production resources to production tasks by using a multiple attribute decision making technique suitable for flexible manufacturing. MADEMA tries to establish a decision making framework that can integrate traditional techniques for process planning with scheduling.

MANUFACTURING SYSTEMS AS A DECISION ENVIRONMENT

In order to effectively apply a decision method to a particular problem, the decision environment has to be clearly defined. In this analysis, the assumption is

made that a manufacturing system can be modelled as an hierarchical structure with distinct levels of control. For example, four such levels can be assumed:

- Factory Level
- Job Shop Level
- Work Center Level
- Work Unit Level

The factory level controls a number of job shops, which in turn manage a number of work centers; work centers consist of work units (human beings, machine tools, handling devices, etc.) and can be configured flexibly or rigidly. Furthermore, a distinction can be made between work centers:

- Containing resources/machines that perform similar operations (e.g. turning) but are not necessarily identical; some of the resources/machines may be interchangeable while others can only process certain types of jobs.

- Containing almost identical resources/machines, all of which are interchangeable.

The approach described in this paper is particularily concerned with the work center level. Here, the goal is to assign resources such as machine tools, handling devices, operators, and robots to production tasks in the most efficient manner. The decision environment at this level is distinguished by two important characteristics:

- Decisions are made repeatedly and in a relatively short period of time.

- Decisions are made from a "risk neutral" standpoint which means that concern is more with the average outcome of a large number of decisions rather than the specific outcome of each individual decision.

Indeed, in the manufacturing environment and at the work center level, numerous decisions about resource assignment must be made in short order. If occasionally a decision is "incorrect", its impact will be relatively insignificant. However, if a series of decisions are systematically wrong, their impact will be of great significance. Such is the standpoint of the methodology presented in this paper.

Although the approach described in the sequel is within the context of the aforementioned hierarchical structure and the associated definitions, its character is general, and it does not require for its implementation a specific control hierarchy.

THE MADEMA APPROACH

The primary goal of a work center control system is to coordinate the effective utilization of the work center resources. As the complexity of the production system increases, the decision making process for production resource assignment becomes more difficult. Traditional decision making methods such as those based on a foreman's rule of thumb or computer simulation may not be adequate to handle such a dynamic environment.

The MADEMA approach is proposed to solve the decision problems related to the assignment of production resources with a multiple critera decision making technique which accommodates the dynamic structure of the decision environment. MADEMA considers a set of alternatives for the execution of a particular production task. The choice of one alternative resource over another is made by an evaluation of relevant criteria/attributes. Thus, a decision matrix can be formed where the rows AL_i ($i = 1,...n$) represent alternatives, the columns AT_j ($j = 1,...m$) represent the attributes, and the entries a are the values of the attributes for the corresponding alternatives (Figure 1).

	AT_1	AT_2	.....	AT_m
AL_1	a_{11}	a_{12}		a_{1m}
AL_2	a_{21}	a_{22}		a_{2m}
⋮				
AL_n	a_{n1}			a_{nm}

Figure 1: Decision Matrix for m Attributes and n Alternatives

MADEMA entails five consecutive steps which must be completed before a decision is reached. These steps are:

- Determine Alternatives (AL_j)

  Decision implies choice, and choice implies alternatives. The set of alternatives considered must be complete enough so as to guarantee inclusion of the best possible decisions. In this discussion, an alternative is defined as a possible set of resource assignments to specific tasks. In order for the alternatives to qualify for consideration, they must satisfy both feasibility and availablity requirements. For example, the machine tools must be capable of producing the part within its specifications and also be available at the required time. This calls for a means to determine which resources have the potential to complete the required task and when these resources will be available. Process planning has traditionally been used to determine the operations sequence and, sometimes, the machine tools and operational data required for parts production. It appears to be a suitable tool for determining the alternatives on the basis of technological consideration.

- Determine Attributes (AT_j)

  Attributes are the criteria according to which the different alternatives are to be evaluated. Criteria such as time, quality, and experience can be considered. The correct determination of attributes is crucial to the effectiveness of any decision method. Artificial intelligence techniques can be used in order to establish a systematic way to deduce and quantify relevant attributes.

- Determine Consequences with Respect to the Attributes for Each Alternative (a_{ij})

  "Consequences" are the values the attributes take on at the time the decisions are made. They are needed for evaluation and, eventually, selection of alternatives. To determine consequences, relevant data must be obtained. Processing time and cost may be derived from the outcome of process planning; job durations can be estimated by Industrial Engineering Time

Standards. Criteria not so easily quantifiable are those such as reliability and experience and can be quantified by a rating based on past history using statistical methods and "fuzzy set" approaches.

- Apply Decision Rule(s) For Choosing the Best Alternative

  A decision rule is thought of as the way to choose the "best" available alternative. For example, a linear combination of attributes with weights that indicate their relative importance to the decision maker can be used to evaluate the alternatives. The decision rule can be the one that chooses the alternative with the greatest likelihood of producing a higher utility value.

- Select the Best Alternative

  Following the setting up of decision rules, the best alternative can then be chosen based on the established rules. For decision making at the work centers, the best alternative can be a set of resource assignments yielding an optimal utility value.

MADEMA AND PROCESS PLANNING

While the establishment of adequate decision rules for resource assignment within a manufacturing system is a crucial task, the determination of a set of feasible alternatives in the form of resource-job assignments is the initial step of the method and thus of great importance. Indeed, resources within the work center must be technologically capable of executing the production tasks before they can be considered as part of any alternative. Process planning has traditionally been employed to determine the operation sequence and the operational data for parts manufacturing. In general, process planning systems tend to designate specific operations to specific machines at relatively early planning stages and thus reduce much of the planning flexibility.

In order to develop alternative resource machines for the various production tasks, an advanced computerized process planning system (XPS-1), developed by CAM-i, has been considered as a typical generative process planning system; an interfacing module, X-MAG (Experimental Machine Alternative Generation), has been

developed for the purpose of interfacing XPS-1 with the proposed MADEMA concept. Although XPS-1 has been used as the process planning system to be interfaced throughout the development of X-MAG, the concept of X-MAG is of general interest and can be applied to any computerized process planning system.

As with many other process planning systems, XPS-1 presently delivers a sequence of work elements without determining the machines on which these work elements can be performed. To determine machine alternatives, appropriate data between part/feature and machines should be compared. The outcome of such a plan will then be a sequence of work elements with a relevant set of machine alternatives.

MADEMA SOFTWARE

In the course of the research work resulting in the MADEMA concept, the above steps were addressed in such a way that an integrated decision making system for work center control is now under develoment. This system should be thought of as an "intelligent work center controller", having the capability of making decisions regarding the assignment of work center resources to production tasks in a very short period of time (in the order of magnitude of seconds or minutes) and, thus, being able to provide the required intelligence in order that decisions can eventually be made automatically without human intervention.

The first steps in developing this "intelligent work center controller" have already been undertaken and have resulted in the development of experimental software. X-DAL, which stands for eXperimental Determination of ALternatives, is the software program which provides the means for determining the alternatives. X-MAG is the program interfacing process planning with MADEMA with the purpose of determining the feasibility of the available resources. X-DETA, an eXperimental software program for the DETermination of Attributes, is a rule-based system which allows the selection of attributes according to work center conditions and characteristics of the jobs to be done. X-EVA is the eXperimental software program which EValuates the Alternatives, assuming that the consequences of each alternative with respect to the determined attributes are known, and entails a new, unique decision making technique that considers the characteristics of the decision environment. Furthermore, in the course of this research work and in order to evaluate the adequacy and practical relevance

of the proposed concepts, an experimental software program for system emulation has been developed under the name X-EMA.

Most of the above-mentioned software run on a VAX 11/750 and are written in NIL (New Implementation of LISP, a publicly available version of LISP). X-DAL is written in Fortran because, as a base for its development, the process planning system XPS-1, already developed by CAM-i, has been used. However, any other process planning system in use by any company can be interfaced with MADEMA.

CONCLUSIONS

A MAnufacturing DEcision MAking (MADEMA) method has been described for the assignment of factory resources to production tasks. The method addresses the issue as a multiple attribute decision making problem and suggests five decision making steps. The concept is adequate for CIM.

PATRIARCH:
HIERARCHICAL PRODUCTION SCHEDULING

Stephen R. Lawrence
Thomas E. Morton

Graduate School of Industrial Administration
Carnegie-Mellon University
Pittsburgh, PA 15213

OBJECTIVES and ATTRIBUTES

Scheduling research has traditionally examined small, well-defined problems that often do not map well to real world scheduling circumstances. PATRIARCH, a joint research project involving Carnegie-Mellon University and IBM, has defined a general hierarchical approach for dealing with the complexity of industrial scheduling environments. The objective of PATRIARCH is to provide an integrated real time production support system to plan, schedule, and dispatch work in a "real world" production setting. The intent of this paper is to present an overview of the PATRIARCH system in general, and to describe the implementation of its component heuristic scheduling algorithm in some detail.

In its full implementation, PATRIARCH will incorporate a combination of heuristic scheduling algorithms, artificial intelligence knowledge representation techniques, and expert rule-based production systems. The component parts of PATRIARCH have been in independent development for a number of years. Morton and collegues have developed a class of heuristic scheduling algorithms [Morton *et al.* 83] which have proven successful in scheduling single machine job-shops [Rachamadugu and Morton] 82], flow-shops [Vepsalainen *et al.* 82], and projects [Lawrence 84]. Fox has represented scheduling knowledge in semantic networks, using constraint directed reasoning to provide feasible schedules in complex factory environments [Fox 83]. More recently, Smith and Ow have used advanced knowledge representation techinques to identify bottleneck resources and produce feasible schedules [Ow and Smith 86]. By bringing these seperate lines of inquiry together into one system, the strengths of each can be exploited, and the shortcomings of each mitigated.

In support of its objective, PATRIARCH exhibits a number of important characteristics. In general, it is intended to conform to the thought processes of human users, and the informantion flows within human organizations. PATRIARCH will support human decision making -- it will not impose decisions. More specifically, PATRIARCH can be defined by describing its attributes:

1. **Hierarchical.** Levels of aggregation within PATRIARCH map to the organizational, physical, and temporal divisions of the firm (see next section). Depending on the decision to be made, different levels of aggregation can be called into play.

2. **Distributed Decisions.** Decision making is retained at local levels. Local decisions are guided by economic and lead time information passed from higher levels of aggregation. In turn, local decision levels pass information up the hierarchy, where it can be used to influence other more remote decisions.

3. **Product Structure Based.** Demand for component parts and subassemblies is derived from finished product demand (both known and forecast) in the manner of MRP bill of material explosions. Unlike MRP, the leadtimes and priorities of component parts and subassemblies are updated dynamically as time progresses and shop conditions change.

4. **Economic Objective.** Rather than imposing an artificial objective such as makespan, or weighted tardiness, on the decision making process, PATRIARCH works to minimize the aggregate Net Present Value (NPV) of its schedules over time. It will be argued that an NPV objective subsumes most other commonly used scheduling objectives.

5. **Dynamic.** As time progresses, demand forecasts are updated, shop floor conditions fluctuate, resource availabilities change, and consequently, the production scheduling task is constantly evolving. PATRIARCH accomodates this evolution by supporting real-time scheduling in response to a dynamically changing environment.

6. **Decision Support Focus.** The expertise of human decision makers is retained, and supported by easy "what-if" capability.

7. **Extensible.** The modular structure of PATRIARCH accomodates easy modification and extension to allow inclusion of lot-sizing, preemption, and inventory planning.

SYSTEM STRUCTURE

Hierarchical Structure

As noted above, the hierarchical structure of PATRIARCH conforms to the organizational, temporal, productive, and planning divisions of the firm as shown in *Exhibit 1.* The four hierarchical levels included in this exhibit are for illustrative purposes only -- any number of levels can be accomodated. For instance, large firms may require a *divisional* level to be interposed between the *firm* and *plant* levels; and very small firms may be able to collapse the *firm*, *plant*, and *work center* levels into a single unit of aggregation.

Data Structures

The ambition of PATRIARCH is to create a generalized scheduling tool which can be applied to a variety of scheduling environments (flow shops, job shops, projects, *et cetera*). Critical to the realization of this ambition is the formulation of a generic scheduling framework which can admit the widest possible variety of scheduling tasks. Within PATRIARCH, scheduling is conceived to be the deployment of resources to satisfy customer demands for the products of the firm. In support of this framework, there exist four primary data structures within PATRIARCH: *resources*, *standards*, *orders*, and *activities*.

Exhibit 1
Hierarchical Structure

	DIVISIONS of the FIRM			
PATRIARCH	Physical	Temporal	Productive	Planning
Level 1	Firm	months/years	lines	strategic plans
Level 2	Plants	weeks/months	products	master schedules
Level 3	Work Centers	days/weeks	assemblies	load profiles
Level 4	Machines	mins/hours/days	parts	dispatch schedules

The generality of this framework allows the representation of a wide variety of scheduling environments including flow shops, job shops, and projects.

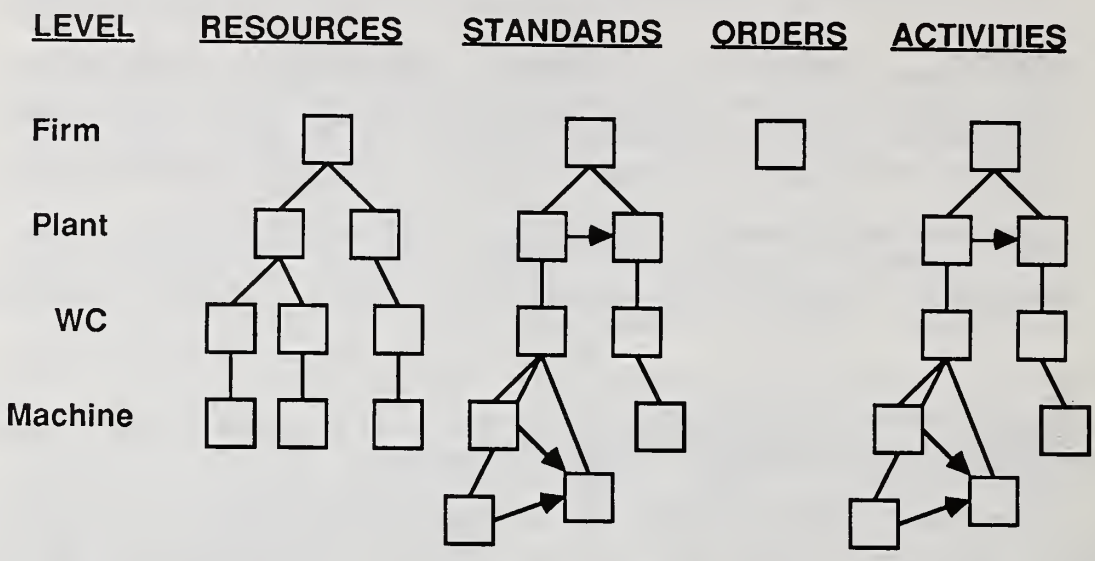
Resources. Resources include all productive assets of the firm including the firm itself, its component plants, work centers, and machines. Resources are deployed by the firm to produce products, assemblies, and parts. In general, many resources are in short supply, constraining the quantity and timing of production -- it is the short supply of resources that gives rise to scheduling problems. Attributes of resources include a *maximum capacity*, and an *open capacity* based on current resource loadings. Resources are hierarchical in nature as illustrated in *Exhibit 2*.

Standards. Standards represent all of the engineering and production data that is required to create properly a product using deployed resources. Attributes of standards include a *bill of materials* representing the component parts of a product, *production standards* based on time estimates of production processes, and *cost data* representing the accounting costs of undertaking a productive task. In addition to a hierarchical structure, standards also exhibit a precedence structure as illustrated in *Exhibit 2*. The precedence structure of standards captures the knowledge that certain tasks must be undertaken and completed prior to the start of subsequent tasks. In addition to standards for physical tasks (such as the "stuffing" of a circuit board), standards can exist for administrative tasks. For instance, prior to manufacture, parts must be ordered from vendors. The tasks of creating a purchase order, waiting for delivery of the ordered material, and the inspection of incoming orders can all have associated standards.

Orders. Orders represent demand for the productive output of the firm. The attributes of orders include *quantity data* representing the amount of end product desired by the customer or client, *prices* representing the revenues to be received by the firm subsequent to the delivery of the order, *timing information* regarding the due date of the order and the terms of payment, and *penalty information* regarding any penalties to be applied in case of non-compliance with delivery terms. Orders are not hierarchical, and represent demands for end products only. Depending on the level of aggregation, orders can be a combination of actual and forecast demands.

Activities. Activities are the product of standards and orders, and represent the tasks that must be undertaken to fullfill orders. An order for a product of the firm causes a cascade of activities to be instantiated as the requirements of the order are traced down

Exhibit 2
Data Structures



through the standards hierarchy. For instance, an order for a large mainframe computer will result in the eventual creation of an activity to manufacture a power supply, among many others. The attributes of activities include *scheduling information* such as processing times, due dates, and quantities; *cost data* representing the expected material, labor, and overhead costs of the activity; and *precedence relations* representing the technological ordering of activities. Activities are hierarchical in nature as illustrated in *Exhibit 2*.

PROBLEM DEFINITION

With a general representation of the scheduling domain established, an explicit characterization of the scheduling problem can be made. In words, the scheduling task is to sequence activities arising from real and forecast demands using deployed resources in such a manner that the Net Present Value of the resulting cash flows is maximized. This scheduling objective must satisfy technological ordering constraints and resource capacity constraints. Expressed in the format of a mathematical programming problem, the scheduling problem defined here can be represented as:

MAXIMIZE Net Present Value of cash flows
from time sequenced activities

Subject to

Resource capacity constraints
Technological precedence constraints

NPV Objective

Cash flows arise when expenses are paid or revenues received. Given an opportunity cost of capital, future cash flows can be reduced to a single Net Present Value in the usual manner. Note that NPV calculations require the use of *cash* inflows and outflows, not changes in incurred obligations such as debt. For instance, the value of a product sale is discounted from the time that the associated invoice is paid, *not* from the time the product is shipped or the time an invoice is issued. Examples of cash inflows and outflows associated with scheduling are included in *Exhibit 3*. Clearly, the NPV of a schedule is increased by delaying cash outflows, and accelerating cash inflows as much as possible.

Exhibit 3
Cash Inflows and Outflows Arising From Schedules

CASH OUTFLOWS	CASH INFLOWS
material costs	progress payments (if any)
labor costs	final payment for order
other direct costs	
inventory holding costs	
one-time tardiness penalties	
accruing tardiness penalties	

While not yet proven analytically, there is emerging evidence that the NPV objective subsumes most other commonly used scheduling objectives. For example, the *makespan* of a schedule is often minimized by maximizing the NPV of activities which have cash inflows payable upon order completion. The *number of tardy orders* is minimized when orders have very large one-time tardiness penalties. The *weighted mean tardiness* of orders is minimized when orders have large accruing tardiness penalties. Maximizing NPV appears to balance a number of traditional scheduling objectives in the best interests of the firm.

Constraints

Constraints on the NPV objective include limited resources and technological precedence constraints. Resource constraints can be classified into two types: *renewable resources* which effectively can be reused continuously (e.g. machines and labor), and *expendible resources* which are depleted with use (e.g. money). PATRIARCH considers only renewable resources. Technological precedence orderings are currently considered to be deterministic within PATRIARCH -- algorithms for rework cycling and alternative routings are under investigation but are not yet implemented.

HEURISTIC PROCEDURE

Simple versions of the scheduling problem described above can be solved to guaranteed optimality using integer or dynamic programming, but for problems of any complexity, it is well known that the combinatorial explosion of feasible sequences makes the possibility of guaranteed optimality remote [Karp 72]. As a consequence, *heuristic* procedures for generating good feasible schedules are required.

The heuristic scheduling algorithm employed within PATRIARCH has five main components: (1) an estimate of scheduling costs, (2) a tardiness cost forecast, (3) a benefit/cost

scheduling heuristic, (4) an iterative estimation of activity lead times, and (5) an estimation of marginal resource values. Each of these are described below. Required notation is defined in *Exhibit 4*.

Scheduling Costs

The cost associated with performing activity i is simply the sum of Present Values (PV's) for material, direct, and marginal resource costs:

$$V_i = B_i + D_i + C_i p_i Z_k \qquad \text{for all } k \text{ in } R_i$$

If the processing of activity i can be delayed for an additional time unit, savings of IV_i, the opportunity cost of scheduling activity i, can be realized.

Forecast Tardiness Costs

Since activities are created in response to orders, each activity is linked to a single order. The due date of an order can therefore be used to calculate a due date for each activity associated with it (in the same manner that CPM project activity due dates are calculated). Consequently, a delay in any activity past its due date will cause the order to be late with consequent tardiness costs. Hence, the subscripts j in Exhibit 4 can be replaced by subscripts i when considering individual activities.

If activity i is past due by X days, total tardiness costs incurred are the sum of the Present Values for the tardiness penalties, the opportunity costs of lost revenues, and the cost of lost customer goodwill:

$$W_i = TF_i + X(TA_i + IR_i + G_i) \qquad \text{when activity i is past due}$$

If the activity is delayed an additional day, the marginal tardiness costs are:

$$w_i = TA_i + IR_i + G_i \qquad \text{when activity i is past due}$$

If activity i is waiting for access to resource k, and is *not* past due, no tardiness penalty will be incurred by delaying activity i by one time unit. However, if a higher priority activity arrives in the queue for resource k, the start of activity i will be further delayed, possibly making it past due with consequent tardiness costs. Several researchers have developed a method of heuristically estimating such tardiness costs [Morton *et al:* 83], and have found that increasing forecast tardiness costs exponentially over time for *slack* activities provides excellent results:

$$w_i = (TF_i + TA_i + IR_i + G_i) \exp[-s_i/(2p_{ave})]$$

$$= W_i \exp[-s_i/(2p_{ave})]$$

where p_{ave} = average processing time of all activities

s_i = slack of activity i

Note that this expression applies *only* to activities that are not past due.

Benefit/Cost Heuristic

The core of the PATRIARCH heuristic is to compare the immediate benefits of delaying the start of an activity with consequent downstream tardiness costs. The *delay cost* of an

Exhibit 4
Notation

$\mathbf{R}$ = set of resources deployed by the firm

$\mathbf{S}$ = set of standards defining the products of the firm

$\mathbf{O}$ = set of actual and forecast orders for products

$\mathbf{A}$ = set of activities instantiated from standards by orders

I = opportunity cost of capital to the firm per period

t = current time

B_i = PV of bill of material costs incurred by activity i

D_i = PV of other direct costs (e.g. labor) incurred by activity i

p_i = processing time of activity i

d_i = activity due date

s_i = activity slack = $\max[0, d_i - t]$

$\mathbf{R}_i$ = subset of resources required by activity i

$\mathbf{C}_i$ = set of % capacity requirements for each element of $\mathbf{R}_i$

O_i = order associated with activity i

V_i = PV of total cost of scheduling activity i

U_i = PV of cost of delaying activity i by one time unit

RR_i = PV of rate of return expected from scheduling activity i

P_j = PV of payments received upon completion of order j

PT_j = terms of payment

TA_j = PV of accruing tardiness penalty if order j is late

TF_j = PV of fixed one-time tardiness penalty if order j is late

G_j = PV of dollar estimate of lost goodwill if order j is late

W_j = PV of total forecast tardiness costs of order j

w_j = PV of marginal forecast tardiness costs of order j

Z_k = PV of imputed marginal value of resource k

activity is the difference between the opportunity cost of immediately starting the activity and the forecast tardiness cost if the start of the activity is delayed by one time unit:

$$U_i = w_i - IV_i$$

A rate of return for immediately scheduling the activity is similarly calculated:

$$RR_i = (w_i / IV_i) - 1.0$$

These definitions immediately suggest a *sequencing decision rule.* Consider a number of activities waiting for access to an occupied resource (e.g. a work center). *Dispatch* activities to the resource as follows:

1. Wait until the resource has capacity to accept an additional activity.

2. Release that activity in the queue with the maximum forecast rate of return.

3. If *no* activity has a positive rate of return, schedule *no* activity.

With respect to the third rule, if the benefit of delaying the release of an *order* is greater than its cost, then the resource can be better used to run a more critical order, perhaps not yet in the resource queue. Whether or not the component *activities* of released orders should be released in a similar manner is under investigation -- there is evidence that activities should be released as soon as resources become available.

Lead Time Iteration

One difficulty in estimating the due dates for activities is in estimating the time they will wait for resources to become free. For heavily used resources, this time-in-queue is significant and must be considered when calculating the due dates for activities. A means of iteratively estimating these activity lead times has recently been developed [Vepsalainen *et al.* 82]. In the first iteration, activity slacks are calculated considering only the processing times of downstream activities. Activities are then scheduled using the benefit/cost heuristic (above) with these initial activity slack estimates. The amount of time each activity i spends waiting for resources in the initial schedule is recorded as the queue time of that activity. In subsequent iterations, activity slack is estimated using the sum of activity processing and queue times:

$$s_i^n = \max\{ 0, d_i^{n-1} - p_i - q_i^{n-1} \}$$

where $q_i^0 = 0$

n = iteration number

Experience indicates that 3 or 4 iterations is sufficient to develop good activity lead time estimates, resulting in improved schedules.

Marginal Resource Values

The final component of the PATRIARCH scheduling heuristic is estimating the marginal values (prices) of resources. This is accomplished by noting that if resource k is shut down for one time period, the cost of the shutdown will be approximately equal to the sum of the delay costs for all activities using the resource during the current time period. As in the case of estimating lead times, marginal resource costs are determined iteratively:

$$Z_k^n = SUM_Y \, U_i^{n-1}$$

where Y = set of activities using resource k at time t

n = iteration number

Calculations for lead times and marginal resource values are carried out simultaneously in the iterative process.

Exhibit 5
Scheduling Example

Cost of capital = 26% / year
Time periods / year = 52 weeks / year
Period capital costs = 0.5% / week

Activity Data

ACTIVITY (i)	PROCESS (p_i)	COSTS (V_i)	DUE (d_i)	PRICE (P_i)	TERMS (N_i)
1	1	6	2	10	2
5	5	30	7	50	3
3	3	18	8	30	1
9	9	56	40	90	4
7	7	21	11	70	2

Results

HEURISTIC	SEQUENCE	MAKESPAN	NPV
FCFS	1-2-3-4-5	25	76.45
SPT	1-3-5-2-4	25	78.99
EDD	1-2-3-5-4	25	79.32
MST	1-2-5-3-4	25	79.31
PATRIARCH	1-2-3-5-4	36	82.02

NOTE: PATRIARCH heuristic delays start of activity 4 until time t=25; other heuristics schedule activity 4 as soon as resource becomes available.

A SMALL EXAMPLE

The functioning of the PATRIARCH heuristic is probably best illustrated with an example. Consider 5 activities that are queued in front of a resource (say a work center). Each activity will require the entire capacity of the work center. No tardiness penalties are incurred if an activity is late, other than a delay in the receipt of revenues. In this example, assume that lead times and resource marginal costs have been iteratively

estimated, and a final schedule is to be determined. The firm uses an annual opportunity cost of capital of 26%, and the appropriate time period for the current level of aggregation is one week. The objective is to release the activities to the work center so that the NPV of the resulting schedule is minimized.

Specific information for each activity is given in *Exhibit 5*, along with the resulting schedules using PATRIARCH, and several other popular scheduling heuristics. As illustrated, the PATRIARCH heuristic provides a schedule with a superior Net Present Value -- if the NPV's are in units of $1000, the PATRIARCH schedule is worth $2,700 more to the firm than any other schedule.

PATRIARCH succeeds in creating a better schedule by delaying the release of activity 4 until time t=25. How this happens is illustrated in *Exhibit 6*. Note that PATRIARCH delays the release of activity 4 until the forecast tardiness cost of further delay exceeds the marginal scheduling cost. This delays cash outflows, and consequently increases the NPV of the resulting schedule.

CONCLUSIONS

PATRIARCH is an evolving implementation of a general hierarchical framework for coping with complex industrial scheduling environments. This framework conceptualizes the scheduling environment to be comprised of *resources*, *standards*, *orders*, and *activities*. The scheduling problem is to time sequence activities in such a manner that the Net Present Value of the resulting schedule is maximized, subject to resource and technolgical precedence constraints. In its full implementation, PATRIARCH will address the complexity of real-world scheduling problems by incorporating a combination of heuristic scheduling algorithms, AI knowledge representation techniques, and expert production systems.

Exhibit 6
PATRIARCH Scheduling Detail

Detail of PATRIARCH schedule for activity 4:

Time (t)	Slack ($s2_i$)	Marginal Scheduling Cost (IV_4)	Forecast Tardiness Cost (w_4)	Rate of Return (RR_4)
0	31	0.28	0.02	-0.93
10	21	0.28	0.06	-0.79
20	11	0.28	0.15	-0.46
26	5	0.28	0.27	-0.04
27	4	0.28	0.30	**+0.07**

Note that PATRIARCH delays scheduling activity 4 until the forecast tardiness cost exceeds the marginal scheduling cost.

The heuristic scheduling algorithm used within PATRIARCH works by comparing scheduling costs with tardiness costs. Only when *forecast* tardiness costs exceed *marginal* scheduling costs is an activity considered for scheduling. In this manner, cash ouflows are delayed, and cash inflows accelerated as much as possible, thereby increasing the NPV of the resulting schedule. Experience to date shows the PATRIARCH heuristic to provide schedules with NPV's greater than those of other common scheduling heuristics. In addition

to more fully testing the PATRIARCH heuristic, a number of other research issues must be addressed in the future. These include lot-sizing issues, alternative routings of work, rework cycles, and sequence dependent setup times.

REFERENCES

[Fox 83]
Constraint-Directed Search: A Case Study of Job-Shop Scheduling,
PhD Thesis, Carnegie-Mellon University, 1983.

[Karp 72]
Karp, R.M.
Reducibility among combinatorial problems, in *Complexity of Computer Computations*
Miller, R.E., and J.W. Thatcher (editors), pages 83-103, Plenum Press, 1972.

[Lawrence 85]
Lawrence, S.R.
Heuristic algorithms for resource constrained project scheduling,
unpublished report, GSIA, Carnegie-Mellon University, 1985.

[Morton *et al.* 83]
Morton, T.E., R.M. Rachamadugu, and A.P. Vepsalainen,
Accurate myopic heuristic for Tardiness Scheduling,
Working Paper W.P. 15-83-84, GSIA, Carnegie-Mellon University, 1983.

[Ow and Smith 86]
Ow, P.S., and S.F. Smith,
Towards an opportunistic scheduling system,
in *Proceedings of the Hawaii International Conference on Systems Sciences,*
Honolulu, Hawaii, 1986.

[Rachamadugu and Morton 82]
Rachamadugu, A.P., and T.E. Morton,
Myopic heuristic for the single machine weighted tardiness problem,
Working Paper 28-81-82, GSIA, Carnegie-Mellon University, 1982.

[Vepsalainen *et al.* 82]
Vepsalainen, A.P., R.V. Rachamadugu, and T.E. Morton,
Lead time iteration in flow shop scheduling,
Working Paper, GSIA, Carnegie-Mellon University, 1982.

Low-Level Interactive Scheduling

Richard Conway and William Maxwell

Cornell University
Upson Hall
Ithaca, N.Y. 14853

The scheduling function in manufacturing can be usefully partitioned into two stages:

a. planning, in which future decisions are made on a tentative basis

b. action, in which an imminent decision is committed and implemented.

The planning stage, itself, is often further partitioned into stages, and "action" could just be considered the last of these stages. However, the action stage is distinctly different in several important ways:

1. Scheduling is an increasing information and decreasing uncertainty process: planning is based on incomplete and uncertain information; the action decision is based on more and better information.

2. The cost of change of a scheduling decision increases sharply as you go from planning to action. Presumably, action implies a commitment of physical resources,and subsequent change -- if such is possible at all -- may well involve substantial costs of scrap or rework.

It is a curious fact that the action stage of production scheduling has enjoyed relatively little benefit from contemporary computer technology. The final decision to initiate a particular operation on a particular machine is still being made in essentially the same way it was made ten or more years ago. There has presumably been some improvement in the preparatory planning stages, so the set of operations from which the final choice is to be made may be more rationally determined today than would have been the case a decade ago. There has also presumably been significant improvement in the management of materials. But the sad fact is that the low-level, finite-capacity, final-commitment scheduling function has not yet been significantly improved.

In most places today, "scheduling" is understood to mean only the "planning" stages of the process. It is an MRP world, and the word "planning" is explicit in the title. It is, of course, possible that the planning is so effective that the action decision is either automatic or trivially obvious, but we strongly doubt that that is the case. Although there is some effort being made to reduce the length of the period of the planning process, and to feedback some forms of status information into the planning process, neither of these steps is likely to make any major improvement in the process.

There remain two fundamental characteristics of the current planning process that no amount of evolutionary improvement is likely to change:

1. Planning operates with little or no status information. It does not explicitly recognize that the quantity of available resources is limited and highly dynamic: machines break, tools wear, people fail to show up, and material quantity and quality varies from what was expected.

2. Lacking detailed information as to how the final action choices will be made, the planning stage compensates by building-in relatively enormous time allowances for "queuing".

One could imagine extending an MRP system to do detailed, finite-capacity, imminent-action scheduling, but it takes a very lively imagination. Charitably speaking, these systems are already cumbersome, and extending their scope while also converting them to continuous, real-time operation seems somewhat unrealistic.

Scheduling is inherently an exceedingly complex process. There are simply too many variables, and too many possible solutions, for there to be any hope of obtaining optimal solutions to any non-trivial scheduling problems. Anyone who claims otherwise either does not understand the problem, or is being carefully deceptive in the use of the term. However, complexity notwithstanding, scheduling problems are regularly and routinely "solved", since operations do get performed. The point is that if one accepts the challenge as that of improving upon existing practice, rather than assuring a solution that is in some sense optimal, then prospects are much brighter.

A Decision Support Approach to Action-Commitment Scheduling

Our premise is the following:

> Action-commitment scheduling is a continuous, local, manual activity, that takes place within a context that is established by a periodic, remote, computer-based planning process.

Our conjecture is that it should be possible to use modern computing technology to assist in this final stage of the scheduling process. This hardly seems like a debatable proposition, but there are some practical questions:

1. What magnitude of improvement in performance might be obtained?

2. What scale of computing equipment would be required?

3. What is the minimum level of expertise that would be required to operate the system?

4. How would the system "cooperate" with other pieces of the scheduling process already in place? (Only true academics can envision monolithic, "clean-sheet" solutions to the production control

problem.)

5. How difficult would implementation be?

Implicit in this view of the problem is the assumption that the support system would be continuously available and provide essentially immediate response (a few seconds, at most) so that scheduling decisions could reflect absolutely accurate and current status information, and could be made and revised as needed.

We set out to study these issues constructively -- by designing and implementing such a system. This paper is essentially a progress report on that work. The work has been done in collaboration with the Manufacturing Research Center of Hewlett Packard Laboratories, but the conclusions and opinions expressed below are our own and HP should not be blamed.

A prototype system, called LLISS, has been constructed and will soon be subjected to fieldtest. The paragraphs below give a rough description of how LLISS works, and give our preliminary guesses as to the answers to the questions above. But, of course, until there has been some real test in service, the answer to the critical "how much improvement" question remains speculation.

An Electronic Gantt Chart

LLISS is, in effect, simply a contemporary implementation of Henry Gantt's elegantly simple way of organizing and presenting scheduling information. Using a computer for this task is hardly a new idea, but we are unaware of any implementation of the idea that really does justice to it. Today -- two years into this project -- we understand much better why that is the case. First, it isn't easy, and second, computing technology is just now reaching the point where this is a practical, economic possibility.

We envisioned a system that would let the user solve the scheduling problem, by assisting in

1. organizing the massive amount of information that is involved, and permitting selective retrieval and display of that information;

2. providing automatic communication with the external entities that supply this information, and those that use the results of the process; and

3. predicting the implication of each individual scheduling decision.

Of course, any such system capable of supporting completely manual scheduling could be readily enriched with tools for semi-automatic scheduling, but it was never our intention to produce a system that could operate effectively without human intervention.

The Supporting Players

The external agencies that must communicate with LLISS in order to provide the information necessary for action scheduling, and to receive information regarding the results of scheduling, have been modeled as follows:

Manufacturing Engineering

-- Specify the resources ("machines") that are available to perform the work. "Machine" is, of course, a generic term that can represent a machine cell, a person, another department, etc. There are "types" of resource, and each machine is an instance of a particular type. All machines of a particular type are considered interchangeable by the Scheduler.

-- Specify the types of work that the scheduling unit can perform. Each type of work is called a "part", and production of a part is accomplished by performing a sequence of "operations".

Production Planning

-- Specify "tasks" that are to be performed. Each task is a request for a certain quantity of a particular part to be available by a certain time.

-- Learn (from LLISS) when a task is expected to be completed (and when it actually is completed) to coordinate tasks in different scheduling units.

This represents the LLISS interface with the higher-level production planning stages. It isolates LLISS from the questions of explosion, common part aggregation, netting against inventory, and lot size determination.

Conversely, LLISS can supply to the planning stage a "very good" estimate of the time that will be required to accomplish the task -- obviating the need to use a crude and conservative "queuing allowance".

Material Control

-- To learn from LLISS the schedule of "raw material" requirements. That is, how much of what is required when to meet the schedule.

-- To specify the availability of raw materials.

When tasks assigned to LLISS represent the starting point in the production of a product, Material Control represents the interface to purchasing, receiving and inventory. When tasks represent later stages, following the completion of tasks assigned to other units, then Material Control represents another interface to Production Planning. In either event, it separates from LLISS the problems of ordering, moving, and stocking material that is external to the single unit LLISS is scheduling.

Maintenance Control

-- Specify the intervals during which the machines are not available for production, either because of scheduled maintenance or for repair of an unanticipated breakdown.

Essentially, Maintenance Control represents the scheduling of a second set of finite resources ("repairmen") on top of the basic production schedule. Maintenance intervals on the machines take priority over production operations, and pre-empt conflicting production operations.

Machine Control

-- Specify the completion of an individual operation, including the quantity of (good) pieces completed.

-- Specify that a particular machine has "broken down" and is not available for scheduling (until Maintenance has repaired it).

-- To learn from LLISS what operations are "startable" on a particular machine, and specify what selection is to be made.

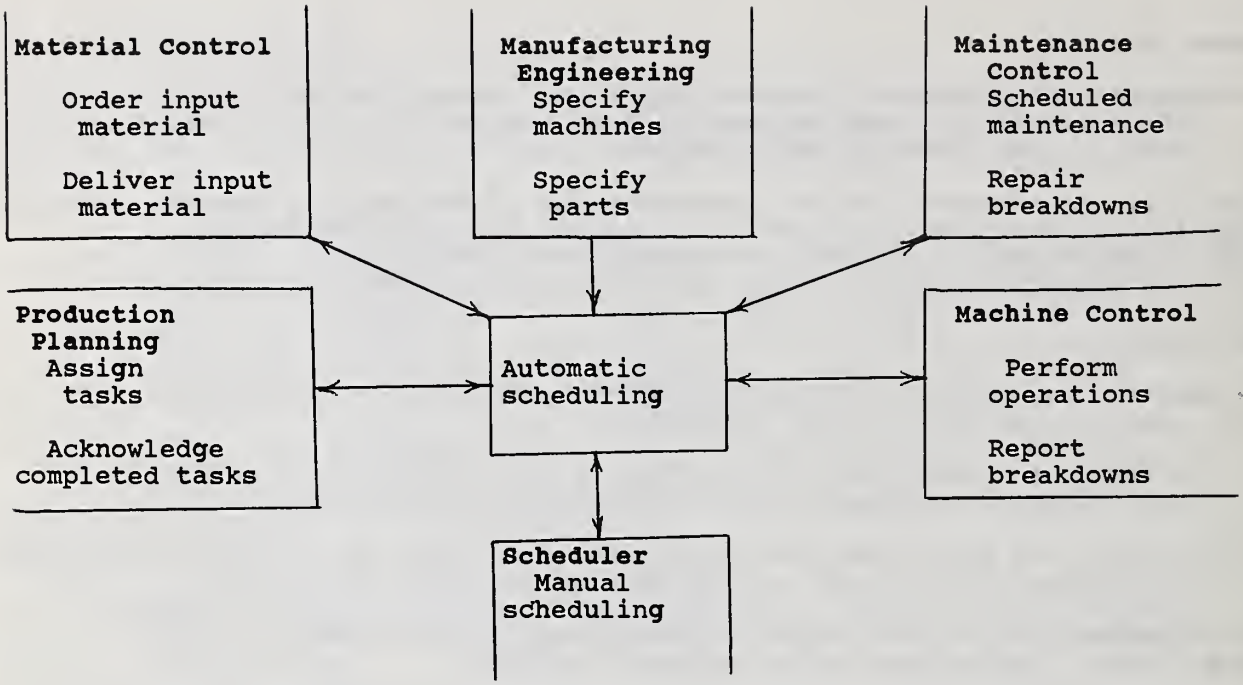
This represents the interface with a human operator of a machine, or with a "level 1 controller" of an automatic machine.

In the current LLISS prototype, each of these supporting roles is served by a special user interface. In each case, all information must be entered manually from the keyboard, and results must be obtained visually from the screen or printed output. There is no direct communication facility. This is, of course, a temporary expedient to permit the development of the prototype in a way that is highly independent of the details of the other information systems in the environment in which LLISS will be tested.

The Primary User: The Scheduler

The human who is responsible for scheduling the local unit is the primary user, and intended beneficiary, of LLISS. It is presumably his task that LLISS is intended to support. Yet, ironically, the Scheduler is the one user whose actions are elective -- LLISS will work without any intervention by the human scheduler, but it cannot work without the specification of tasks (by Production Planning), the delivery of material (by Material Control), the performance of operations (by Machine Control), or the repair of breakdowns (by Maintenance Control).

The Scheduler can establish the ground rules under which LLISS operates, and then let the process operate more or less automatically, intervening only when he thinks he can improve upon things. Schematically, the system works as shown below:



Underlying the whole process is "the schedule". This stretches from "now" indefinitely into the future, and covers all the machines in the unit. The visible Gantt Chart, as shown below, is simply a window that displays some selected portion of the schedule. As various users supply information, the schedule changes automatically. Changes that affect the displayed portion are immediately modified on the screen.

The human Scheduler has a variety of tools with which to modify the underlying schedule. At the lowest level these allow individual operations in the displayed portion of the schedule to be moved. (On an HP machine this can be done directly on the screen using the "Touchscreen" feature. Less well-endowed machines require the use of cursor keys or a mouse.)

There are also higher-level tools that allow the entire schedule to be compressed, re-scheduled by priority, re-scheduled by due-date, re-scheduled by lateness, etc. There are also tools to modify the tasks by task splitting, task combination, quantity reduction, priority change, etc.

The Scheduler has a large number of different ways to view the schedule -- projections by time, machine, task, or status.

We regard LLISS as open-ended with regard both to the tools available to the Scheduler, and the ways in which the schedule can be viewed. Presumably, the initial fieldtest will result in numerous additions of both kinds.

Assignment of a New Task

The LLISS dialog by which the Production Planning user assigns a new task to the scheduling unit is unusual and merits a brief description.

The new-task-assignment dialog is based on a screen like the one shown below. The Planner proposes a specific form of the prospective task -- say ten pieces of PART1 required by production time 100, as shown in column 1 below:

In this example, LLISS has responded that, to achieve the stated requirements, nineteen pieces should be started not later than production time 3. Since that starting time is still in the future, LLISS is prepared to commit to on-time completion of that task -- if it is submitted in that form now. The Planner continued to explore alternative versions of the prospective task, and LLISS indicated how each would fare. In one case, the task would require a starting time that is already past, so LLISS was unwilling to make a commitment. However, the Planner can nevertheless submit that version of the task as a "request", which will require manual interventation to achieve on-time completion.

"Commitment" is not an absolute guarantee of on-time completion, since LLISS cannot predict machine breakdowns or many of the other events that happen in shops to render pre-computed schedules infeasible. Rather, commitment is a declaration of intent, and a reservation that future task commitments must observe. The human Scheduler can override the automatic scheduling, and make decisions that jeopardize this commitment, but both he and the Planner are notified immediately whenever this occurs.

Essentially, this dialog can take the place of the "queuing allowance" in an MRP system. Whenever a conventional MRP system would reach for its stored, conservative constant for this purpose, a MRP-LLISS hybrid would find out from the LLISS component exactly how long the task will take, given the tasks previously assigned, the current status of the machines, and the standard scheduling scheme begin used.

Simulation of Schedule Alternatives

Some scheduling systems, with similar objectives to LLISS, offer a "simulation mode" that allows the system to "run ahead in time" to predict the outcome of alternative scheduling decisions. It is interesting that no such mode is necessary in LLISS because the underlying schedule corresponds exactly to such a simulation -- except that since it always exists, and is updated in real-time to reflect every change in status, load or decision, it is never necessary to enter a special prediction mode.

You can, of course, at any point store the current status of the system in order to explore alternatives. You then choose which version to restore to continue real operation.

Conclusions

The following are preliminary, tentative, and well-mixed with wishful thinking. Nevertheless:

1. Local, interactive scheduling, in the terms described above, is practical today. Problems of reasonable size can be handled with tolerable response times on microprocessors sufficiently inexpensive that they could be used in each unit, and dedicated to this particular job.

For example, one of the Intel 80286 systems (HP Vectra, IBM AT), currently costing approximately $7000, seems quite capable of providing several-second response for a unit of 30-60 machines, and a current load of 100-200 tasks. 640K of main memory, 20M of disk storage, and 400 by 400 points of (color) screen are reasonably adequate to the task.

For larger dimensions, better response, or better display, the Motorola 68020 systems (HP300, Sun, Apollo), at twice the price, offer several times the computing capacity. It seems likely that by the time scheduling systems such as LLISS are really ready for service, microprocessor systems such as these will be available for less than $10,000.

2. The critical limitation of the computing hardware is the graphical display. Machines with 400 by 400 resolution on a 12 inch screen cannot successfully display a single Gantt Chart window with more than 200-300 blocks. (Several colors are necessary for a display of this density to be usable.) 1000 by 1000, 19 inch screens are four times as effective, but unfortunately as also significantly more costly.

This suggests that the effectiveness of an interactive scheduling system will depend importantly on the care with which the system is designed to exploit the available display resources. Overlay windows, variable scope and scale, and every other trick of the craft can be usefully employed in this application.

Some other scheduling systems, with objectives similar to those of LLISS, have elected to use character displays rather than color graphics. While such a compromise was perhaps necessary in the era of 200 vertical point screens, it is no longer, and we doubt that such systems can survive comparison to those with true graphical displays.

3. The really difficult and interesting problem is the management of the user-command interface. With something on the order of a hundred commands to make available to the scheduler, it is a real challenge to provide a self-prompting (menu-oriented) interface with minimal reliance on either the user's memory or an external reference manual. The LLISS prototype suggests that it is possible, but we haven't got it quite right yet.

4. The real issue is, of course, the performance improvement that systems such as LLISS might permit in reduction of manufacturing lead-time, and the resulting reduction in work-in-process inventory. With no experience or data to support the point, we cannot see how even a crude form of LLISS would not result in a major improvement over current MRP practice. In fact, the prospects of complementing MRP systems with LLISS-like, local, action-commitment front-ends should significantly extend the MRP era.

April 1, 1986
National Bureau of Standards
Symposium on "Real Time Optimization for
Automated Manufacturing Facilities"

THE GENERAL EMPLOYEE SCHEDULING PROBLEM: AN EFFECTIVE LARGE SCALE SOLUTION APPROACH

Fred Glover
Center for Applied Artificial Intelligence
Graduate School of Business
University of Colorado
Boulder, CO 80309-0419

Randy Glover
Management Science Software Systems
1040 Lehigh
Boulder, CO 80303

Claude McMillan
Center for Applied Artificial Intelligence
Graduate School of Business
University of Colorado
Boulder, CO 80309-0419

INTRODUCTION

Employee scheduling problems arise in a variety of service delivery settings, including the scheduling of nurses in hospitals, check encoders in banks, airline and hotel reservation personnel, telephone operators, patrol officers, and others. In their simplest form, these problems involve only the assignment of days-off, as in some of the less complex settings for the scheduling of nurses. A typical problem of this form requires the scheduler to give appropriate days off to each of a number of employees who work standard shifts with differing start times while assuring that the required number of employees are on duty throughout the day and week. Variations of this type were addressed in [2] and [10].

The shift scheduling problem, as in the scheduling of telephone operators, is more complex. In shift scheduling, the scheduler works with part-time as well as full-time employees, and shift types contrast with each other in the following attributes:

1) Duration (length)

2) Start times

3) The number of breaks (reliefs)

4) The placement of the breaks

The scheduler must determine the shift types (and the number of each type to employ), and in some cases determine which employee should receive which set of shifts. Union rules and company policy restrictions are handled in a limited fashion. Variations on this problem have been addressed in [3,8,9,10,12,13,14].

The objective in the shift scheduling problem generally is to approximate as closely as possible the desired number of employees on duty, either by minimizing the overage or minimizing the 'shortage/overage' mix.

The more complex days-off and shift scheduling problem [13] comes a step closer to the **General Employee Scheduling Problem**, and some variations [3,8,14] consider also

the assignment of shifts to actual employees, but deal with that assignment of shifts to employees only after the shift types (and number of each) have been determined.

Table I classifies shift scheduling problems as they are described in the literature, and indicates the differences in: problem size and complexity, the approach employed, and the extent to which the approach was actually being used in industry.

THE GENERAL EMPLOYEE SCHEDULING PROBLEM

The more complex scheduling problem which we address has wide applicability, especially in the supermarket, reservation office, and fast food fields. It differs rather dramatically from the days-off and the shift scheduling problems by including important real world features that resist practical solution by methods of formal analysis. We first describe the problem informally and indicate the features that a practical solution system must have in order to deal with the problem effectively. For comparative reference, we indicate overlaps and contrasts with other employee scheduling problems previously examined. We report the result of applying our approach to problems from real world settings and discuss the implications of our empirical results.

FULL-TIME/PART-TIME EMPLOYEES

As in the shift scheduling problem, it is assumed in the General Employee Scheduling Problem that some fraction of the work will be done by full-time employees and the remainder by part-time employees. Full-time employees are those entitled to work a standard number of hours each week (commonly 40) and generally work shifts of standard duration (commonly 8 hours each). The start times and the number and location of breaks may vary. In addition, a full-time employee may or may not be entitled to the same start time each day worked.

TABLE I

A CLASSIFICATION OF THE EMPLOYEE SCHEDULING PROBLEMS
ADDRESSED BY SELECTED PAPERS SINCE 1972

Complexity/Size (for papers cited in References)		Approach Employed		Time Periods	Extent of Use
LOW:					
[2]	U.	formula	Analytic	14	none
[8]	100	shifts	LP & heuristics	32	none
[9]	100	shifts	Branch & bound	72	none
[11]	U.	formula	Analytic	21	none
[1]	U.	formula	Analytic of 7	Cycles	none
MEDIUM:					
[13]	168	shifts	LP 168	none	
[12]	360	shifts	IP & heuristics	49	banking
[14]	300 to 400		Network & heuristics	48	phone co.
[3]	< 500	shifts	Heuristics 48	phone co.	
[10]	< 500	shifts	LP & heuristics	96	phone co.
HIGH:					
Ours	> 1,000,000 shifts		Heuristics - 540 blend of MS/AI	super mkt	fast food

Definitions:

LOW means: No linking constraints between blocks of time periods, a small number of shift types, and homogeneous employees with unlimited availabilities.

MEDIUM means: No linking constraints between blocks of time, a small number of shift types, homogeneous employees with unlimited availabilities, and management rules.

HIGH means: Linking constraints between blocks of time periods, a large number of shift types, non-homogeneous employees with limited availabilities, and management rules.

U. formula (uniform formula) means: Each shift is characterized by the same rule, such as: every employee works 5 days a week (time periods are in days).

The word 'shift' in the foregoing table translates into the word 'variable' in an integer programming formulation. (Some references use the word 'trick' rather than 'shift.')

In the general problem, as handled by our method, the user is given the ability to specify shift features such as duration, start time, and the number and placement of breaks. The user is also able to specify the number of days off and whether they should be consecutive, and to change those features as the system is routinely used from week to week.

In addition to the above, each employee can specify (or the scheduler can specify for the employee):

1) Minimum and maximum hours to be worked during the week.

2) Days and hours of availability during the week.

Item 2 above means that each employee has an availability preference, specifying which days of the week and the hours that employee can work. Since an employee's availability changes (in some settings, about 20% of the work force changes their availabilities each week), the scheduler -- human or machine — must be allowed to edit a file of employee availabilities before addressing the composition of a new schedule each week.

Another important feature of the general problem, not treated in the standard scheduling contexts, is that employees are non-homogeneous in ways beyond their availability preferences and thus cannot be treated as interchangeable entities. Employees have differing skill types, skill levels, and status attributes which limit the scheduler's freedom in assigning shifts. These include, for example, the training required to work at various work stations. Since these change from time to time, the system must allow the scheduler to edit each employee's profile data.

In the General Employee Scheduling Problem, it may also be necessary to observe seniority rules, such as those specifying that employees with more seniority must get more hours of work and start earlier (an early start may be considered desirable), except when other requirements would be violated.

Union/management rules, in the general problem, can require that a specified minimum amount of time must elapse between the time an employee works one day and begins again the next. In some settings they may also require that certain employees, such as students, may not work beyond a specified hour more than one night during the week.

IMPLICATIONS OF THE NON-HOMOGENEOUS EMPLOYEE POOL

In the simpler shift scheduling problems, it is possible to design shifts (and to determine the desired number of each shift type) without regard to employee availabilities, employee skills and skill levels, or employee status. Descriptions of the shift scheduling problem in the literature, on the other hand, sometimes appear to involve a more general solution capability — for example, indicating that shifts are generated to conform to union rules, company policy, etc. What this means, in practice, is that the shifts generated represent categories that are potentially acceptable, but there is no control over whether those selected as a 'solution' have an appropriate composition. This is entirely reasonable for settings where restrictions are loose enough that employees can simply 'sign-up' for whatever schedule is posted, but in broader settings, such a disregard of individual differences can have dire consequences.

In the general scheduling problem, therefore, the design of the shifts and their assignment to specific employees must be coordinated. Designing shifts without considering whether employees can be found to take those shifts will yield either a poor fit (a poor match of employees assigned to employees wanted on duty) or, still worse, an infeasible solution.

LINKING CONSTRAINTS BETWEEN TIME PERIOD BLOCKS

Another significant aspect of the general employee scheduling problem is the existence of linking constraints between time periods. In typical applications of the general problem, each day may be construed as a block consisting of 96 fifteen-minute periods, and the assignment of employees to duty periods during that block is not independent of assignments to employees in other blocks. In addition to restrictions governing admissible assignments on any given day, these linking constraints imply that there are also restrictions governing the total number of periods throughout the entire week, as well as governing the selection of certain types of assignments successively, or cumulatively, across days of the week, e.g., limitations on the selection of opening and closing assignments.

In standard shift scheduling problems, by contrast, blocks such as days are construed as essentially independent. Linking conditions either do not exist or are innocuous enough to be ignored while composing a schedule for any given day (where, for example, the ending conditions for one day may be used to give starting conditions for the next, which again is treated independently).

The days-off and shift scheduling problems can be readily treated as special cases of the general problem, allowing many of the more difficult conditions to be relaxed. By assuming, during the shift design phase, a universe of homogeneous employees without linking constraints across blocks of time, the shift design problem is made relatively simple. Thus a system for the general scheduling problem handles these less restrictive problems as a special case.

THE REAL WORLD SETTING

In real world settings, the manual scheduler (generally a supervisor) works in a highly dynamic mode. Each week, and sometimes more frequently, a new schedule must be created to reflect the altered employee availabilities and changes in the forecasted volume of business. To control costs, management frequently requires that the person-hours of work assigned must not call for a dollar expenditure out of proportion to the dollar sales forecasted.

The manual production of a schedule that respects limited and varying employee availabilities, and yet matches the requirements, is very difficult. The consequence is that the manually produced schedule generally calls for overages (too many on duty) at certain times during the day and week, and — to keep labor costs within acceptable boundaries — produces corresponding shortages (too few on duty) in others. This results in poor service: a condition which managers must seek actively to avoid in highly competitive service industries.

Producing a schedule manually also requires a good deal of time. For example, in the supermarket and fast food industries, our investigations indicate that it takes from 8 to 14 hours for a manager to schedule from 70 to 100 employees for one week, depending on the seriousness devoted to the task. The resulting schedule is likely to be substantially less than optimal. Even when all special conditions may be met (which

is often not the case for schedules produced manually), shortages and overages often combine to yield less than desirable service or an inflated payroll.

IMPLEMENTING OUR APPROACH

To apply our solution method to the General Employee Scheduling Problem, we design shifts with deference to features specified by the user, and with deference to employee availabilities. When a shift is selected, to augment the growing set of shifts which is generated with the goal of meeting the requirements, the identity of the employee to take that shift is specified. This assures that employees are only assigned shifts they are available for. In case the problem lacks a schedule that is feasible in all respects, our approach generates a schedule that nevertheless comes as close as possible to achieving feasibility, then helps the user identify alternative ways to deal with the limitations that created the infeasible situations.

As we subsequently document, our procedure succeeds in solving problems in the range of 1000 times larger than those related scheduling problems previously studied in the literature. Fundamentally, we view our procedure as an integration of management science (MS) and artificial intelligence (AI). Among the levels of procedural generality to which such an integration is relevant, from the micro level of computer coding to the macro level of global strategies, it is the higher levels that have the greatest impact on solution quality and efficiency, and account for what we believe may be unique to our approach.

Evidently, orders of magnitude of difference in the size of combinatorial problems that are successfully treated cannot be explained by clever computer coding. Intermediate level considerations of specific choice rules are more relevant to achieving such successes. The "structure" on which the primary choice rules are superimposed consists of a procedure for building and amending employee shifts which has its roots in the alternating assignment ideas of [5], and which is characterized more broadly in the context of tabu search in [6]. For this application, we conceptually view each stage of generating a partial (or complete) set of duty assignments as creating a trial solution, which is modified or elaborated by transition rules: a standard framework. (Those interested in further details of the system at an intermediate level of implementation are invited to contact the authors.) The key ingredient of our approach lies in the macro level strategies, which combine the perspectives of management science and artificial intelligence in ways not commonly done. At this level, our approach consists of three main components.

First, following an MS based perspective, we develop numerical criteria for evaluating the moves that define possible transitions from one trial solution to another. We do not, however, settle on a single criterion for evaluating a particular type of move. Instead, interlinking criteria and are based on separate evaluation functions are generated which reflect both feasibility and optimality goals. Our use of multiple evaluative criteria presents a new difficulty that MS based heuristics traditionally have not had to face: the fact that a local optimum relative to one criterion may not be a local optimum relative to another. Rather than a stumbling block, we found this difficulty to be a source of fertile opportunity, leading to ways out of blind alleys encountered by other procedures. To exploit this opportunity, we formulated our approach so that each criterion was allowed to "vote" on alternative moves, initially assigning equal weight to the different votes. When a "deadlock" (local optimum) was reached, we increased the weight of those votes that would find a different solution preferable, thereby allowing the procedure to find new trial solutions. The procedure was further endowed with a memory to prevent reversing the direction in which weights

were changed, but allowing the memory to decay so that choices would not be unduly influenced by decisions that should be regarded ancient history. This method for managing memory was implemented by the use of tabu lists as described in [6,7]. Decay factors that "forgot" decisons beyond five to twelve moves earlier all succeeded in avoiding cycling and producing good choices. The combined effect of these features is implicitly to create a tolerance for "bad moves," but only to the extent required to avoid becoming mired down at local optima, resulting in a highly effective strategy.

Our second main component employed an AI based perspective to identify patterns in the ways trial solutions are configured. However, in a departure from standard AI perspective, our goal was not merely to recognize patterns but to create them. In developing this approach, we were guided by the supposition that certain configurations — as manifested in the magnitude and distribution of residual requirements, and available means of meeting them — would ultimately lead to better solutions by our tools of analysis than others. Thus, we adopted the goal of identifying moves that would lead to configurations we judged intuitively promising. This led to creating additional criteria, based on means-end analysis to arrive at exploitable patterns, independent of whether moves to attain these patterns might contribute to the objectives embodied in other criteria. These new criteria were then incorporated into our first component strategy.

The third major component of our approach was to identify significant segments of the problem, and then to subject each segment to its own sequence of solution phases. In this approach, we implicitly perform what might be called a conceptual decomposition. While the problem is an indivisible whole, we nevertheless artficially break it apart. After each round of evaluations and modifications, we put it all back together -- a "Humpty Dumpty" process that characteristically yields gains throughout several repetitions. For example, after a global evaluation of an employee's availability versus needs yet to be filled over all periods (relative to the current state of the solution), bias factors are generated for and against scheduling the employee in particular period blocks. Thereupon, the blocks of time periods are treated as though independent for the purpose of generating the next move. The succeeding global evaluation then fulfills the function of restoring the links between different blocks. A similar procedure is applied to meeting union and seniority rules as the solution process evolves, temporarily decoupling these considerations by bias factors and then restoring them by global review.

Viewing our approach in terms of these macro level strategies, it is clear there is nothing to the AI nor MS oriented approaches that would not, in theory, provide support for our undertaking. In practice, however, AI and MS approaches are usually implemented in a narrower fashion. It is our impression that the approaches most often described in the literature either employ rather shallow "AI/human" intuitive techniques, though sometimes in large numbers, or employ slightly deeper MS heuristic techniques, but in very small numbers. In neither instance do we find great interlinking and overlaying of alternative criteria. Certainly, we have not seen widespread implementation of multiple evaluation functions, integrated and controlled to overcome local optimality and cycling, nor have we seen the active use of pattern creation (in addition to recognition) using notions of exploitability instead of objective gain. Formal mathematical decomposition, although too rigid and inapplicable to discrete problems of the type we examine, has some resemblance to the third component of our approach, if one views the decomposition as susceptible to being carried out in different ways.

THE QUALITY OF THE SCHEDULE

We describe now the results of 10 computer solution tests, producing one schedule for one week, for each of 10 restaurant problems, involving 100 employees, from a real world application in the fast food industry. These tests were conducted on problems selected for benchmark runs by McDonald's Corporation Headquarters, Oak Brook, Illinois. These problems correspond to integer programming problems involving roughly from 1,000,000 to 4,000,000 variables, and from 3,400 to 9,000 constraints, as noted in the integer programming formulation described subsequently. A summary of the test results is provided in the following table.

TABLE II

TEST RESULTS IN A TEN CASE STUDY

Case Number	Number of Employees	Execution Time*	Number of Zero-One+ Variables	Number of Constraints+	Percent Optimality#
1	103	24 min.	2,640,250	3,400	98
2	100	24 min.	1,356,772	3,400	98
3	105	23 min.	1,158,117	3,400	99
4	104	24 min.	3,251,222	4,732	98
5	101	23 min.	2,440,605	4,732	99
6	107	23 min.	4,999,580	4,732	98
7	108	23 min.	1,366,100	4,732	99
8	101	22 min.	4,005,202	4,732	98
9	106	24 min.	2,613,800	9,004	98
10	103	22 min.	1,215,641	9,004	99

*All runs were executed on an IBM PC, 128K memory.

+Determined by program counters, explained in formulation section.

#Solutions verified to be **at least** the indicated percent of optimality.

The percent of optimality figures indicated in the table were arrived at as follows. With each run, no shortages were produced. That is, during no quarter hour period during the week was the number of employees assigned to be on duty less than the desired number. In one run, during 3 quarter-hour periods throughout the week there was an excess of one employee, over and above the number desired. This does not mean that fewer employees could be used, since eliminating an employee would then create shortages in all other 15-minute periods the employee worked (assuming the same employee was on duty in these three 15-minute periods during the week). In two other runs, there was an excess of one employee, during 8 quarter-hour periods during the week, over and above the number desired. For the other seven the overage was between 3 and 8.

In a perfect schedule, the shortages and the overages would all be zero for all periods during the week. In the ten problems tested, periods designated for scheduling equalled 540 (out of 672 15-minute periods for the week). Thus, in the worst case our method achieved zero shortages and zero overages for 98% (532 out of 540) of the total periods under consideration. We did not attempt to obtain the very best solutions possible by our approach (if better solutions did in fact exist), but used an automatic

cut-off rule to terminate search, which accounts for the similarity of run times on problems of different sizes.

In the papers cited that report on an application in a practical setting, comparisons of solution quality are difficult due to different types of objective functions and a frequent lack of explicit performance data. Partial evidence of the quality of performance is offered in [14] by reporting execution time on an IBM 360/67, which ranged from about 40 seconds to something over a minute (the maximum was not specified) for problems with 300 to 400 variables. The other papers, addressing a problem that comes closest to our general problem structure [3] and [10], likewise report reliance on a mainframe computer, but do not offer computation times.

In typical real world settings for fast food stores, (supermarkets, banks, reservation offices, and the like), large mainframes are not common. In fact, it is the increasing prevalence of inexpensive computers that makes automated scheduling in these settings practical.

Our system has accordingly been written in BASIC and implemented on microcomputer systems such as the KAYPRO II, TRS 80 Model II, and the IBM PC. Because our system can handle the less general shift specifications and constraining conditions of the standard shift scheduling problem, we executed experimental runs to determine our execution time for problems whose sizes corresponded to those reported in the literature. For problem profiles corresponding to those in [3,10,14] our solution times on an IBM PC with 128K bytes of central memory did not exceed 50 seconds for problems of up to 500 variables. These **microcomputer** times therefore, in fact, compare favorably to the **mainframe** CPU time reported elsewhere.

It should be noted that solution times for our procedure, using the cutoff rule that achieves the indicated percent of optimality figures, do not increase anywhere near linearly with the number of integer variables. Normally the opposite effect would be expected for discrete problems such as these -- i.e., an increase far worse than linear. This atypical behavior is a key factor reponsible for the advance in the size of real world problems that can be handled successfully.

It is also of some interest to compare the performance of our approach to that achieved for other zero-one problems and not alone for problems closely related to the class we examined. We are prompted to make such a comparison in view of the 1984 Lanchester Prize award for a study of zero-one problems [4]. The citation for the prize underscored the significance of effectively handling these problems as: "Zero-one linear programming is a very important problem in Operations Research. Efforts to solve large problems of this type have continued for over 20 years. However, success has been elusive, and problems with hundreds of zero-one variables and no special structure could usually not be solved in reasonable computation times."

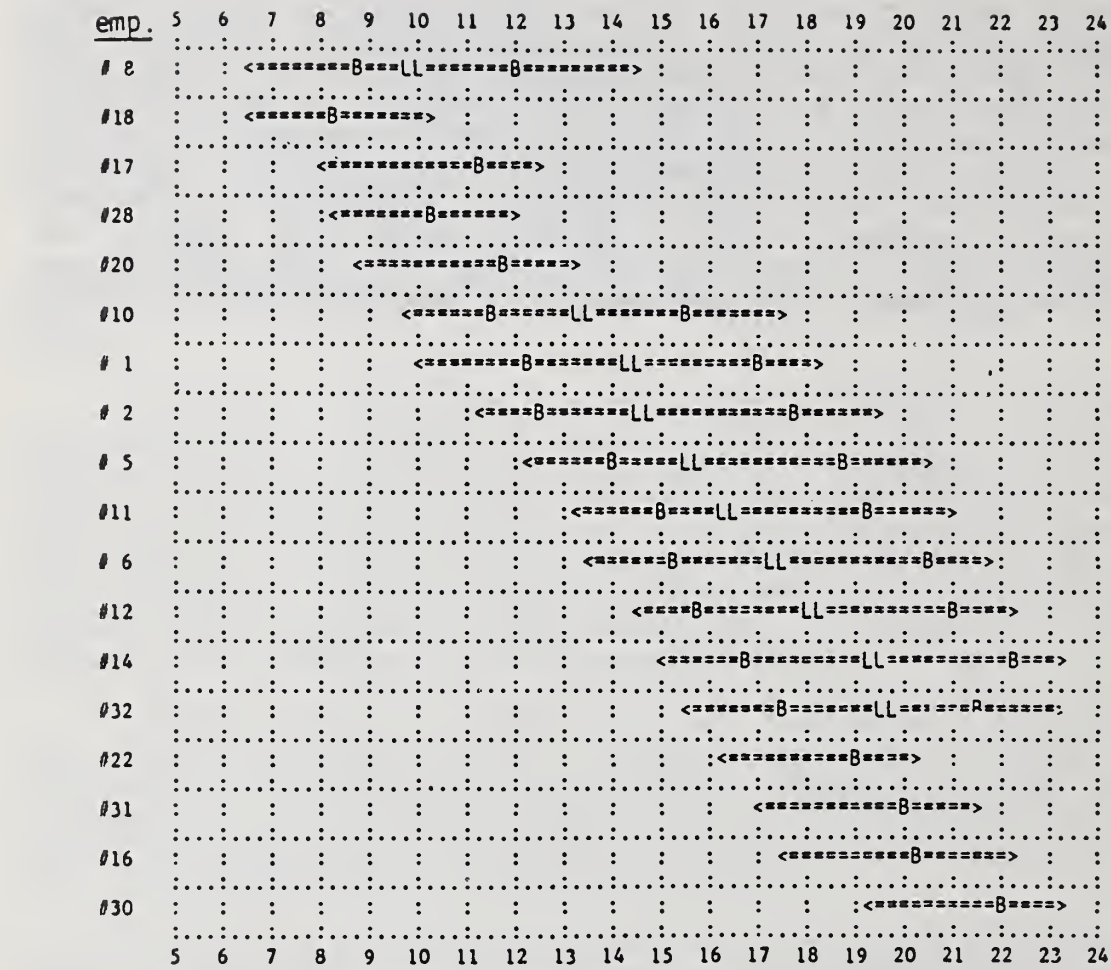
The award-winning study of [4] represents one of the most effective attempts to solve zero-one problems optimally, and provides a major contribution for its ability to perform well on zero-one problems notably larger than those customarily handled. These problems, ten in number, ranged in size from 33 to 2756 zero-one variables. The problems were solved by reliance on a large mainframe computer (an IBM 370/168), and nearly an hour of CPU time was required to handle the largest problem. The contrasting ability to solve problems with 4,000,000 or more zero-one variables to 98% optimality suggests that our approach establishes an appealing trade-off between assured optimality and problem size, and encourages us to believe that gains for solving other zero-one

problems may be possible, perhaps by similar integration of management science and artificial intelligence techniques.

CURRENT USAGE

Our system has been used in several settings for over two years, producing schedules substantially superior to those an experienced scheduler can produce in any amount of time. Applying our system to problems involving approximately 100 employees, the user is able in one to two hours to update the forecast for the coming week, to modify the employee availabilities, and to make "manual" assignments (using the computer) to selected personnel for whom specific work schedules are wanted (e.g., managers). The computer then completes the process by designing and assigning shifts to approximate the optimal match of employees on duty to employees wanted, while respecting the constraints described above. Figure I shows one of the printed outputs which the user may select, indicating by means of a bar graph the shifts assigned to employees on Sunday.

FIGURE I



Schedule for Sunday. In this example the problem size is substantially smaller than in typical applications, both in terms of number of employees and period requirements. At 18:30 it is apparent 10 employees are assigned (#2,5,6,11,12,14,16,22,31 and 32) while 9 are required, yielding an overage of one person for that quarter-hour period. 'B' means 'quarter-hour break', 'LL' means 'half-hour lunch'.

No human intervention is required during the scheduling. Disregarding the improvement in the schedules produced, the process trims 6 to 12 hours off the time required each week for a supervisor to prepare a schedule manually, and this labor savings by itself can pay for the computer in a reasonable period of time.

THE INTEGER PROGRAMMING FORMULATION

A more detailed description of the conditions which were respected in the solutions cited above (the schedules produced using our approach) is as follows:

1) The number of employees on duty (and not taking a lunch or quarter-hour break) in each 15-minute period must come as close as possible to satisfying the demand for employees in that period.

2) All shifts assigned must be members of a feasible set specified by management rules: from 2 to 8 hours duration; with 0 to 2 quarter-hour breaks, and 0 or 1 half-hour break -- depending on the duration of the shift; and with the placement of the breaks specified by reference to "windows" within which they may be moved, as specified by management.

3) No employees can be assigned to more than one shift on any day.

4) No employee can work less than his or her minimum required number of hours during the week, nor more than the maximum.

5) No employee will work less than his or her minimum desired (as opposed to required) number of hours during the week unless there is not enough work to go around. In this latter case, the desired minimum is a goal where employees with greater seniorities have their goals respected first.

6) No employee can work over 6 shifts during the week.

7) Closing and opening rules:

   a) No employee can "close" more than 2 nights in a week, and never two nights in succession (closing means working later than a specified hour).

   b) No employee can close one night and "open" the next day (opening means working earlier than a specified hour).

   c) No student can close more than once during the days Sunday through Thursday.

8) Other things being equal, employees with more seniority must be assured more work and an earlier start.

9) During certain periods of the day those on duty must represent a specified minimal set of skills and skill levels.

The Constraints:

We define:

S = the set of acceptable shift types

S_{ed} = the set of shifts which employee **e** is available to take on day **d**

P_s = the number of quarter-hour periods in shift **s**

x_{esd} = 1 if employee **e** is assigned shift **s** on day **d**;

0 otherwise (decision variable)

A_{ps} = 1 if period **p** is a duty period for a schedule **s**;

0 otherwise

D_{pd} = demand, in projected numbers of employees required to be on duty in period **p** of day **d**

u_{pd} = goal programming deviation variable for falling short of the projected demand

v_{pd} = goal programming deviation variable for exceeding the projected demand

U_e = maximum quarter-hour periods of work for employee **e** during the week

L_e = minimum quarter-hour periods of work for employee **e** during the week

G_e = the desired (goal) minimum quarter-hour periods of work for employee **e** during the week

y_e = a goal programming deviation variable that allows employee **e** to work less than G_e periods if there is not enough work to go around

Then the preceding conditions can be modeled by the following constraints, which are numbered to provide a direct correspondence.

$$\sum_e \sum_{s\ S_{ed}} A_{ps}x_{esd} + u_{pd} - v_{pd} = D_{pd} \text{ for all p and d} \qquad (1)$$

The composition of S_{ed} assures that condition 2) above is satisfied, independently of the other constraints. Therefore, we do not include a corresponding constraint (2) in our formulation.

$$\sum_{s \in S_{ed}} x_{esd} \leq 1 \quad \text{for all e and dS} \qquad (3)$$

$$L_e \leq \sum_{d} \sum_{s \in S_{ed}} P_s x_{esd} \leq U_e \quad \text{for all e} \qquad (4)$$

$$G_e \leq \sum_{d} \sum_{s \in S_{ed}} P_s x_{esd} + y_e \quad \text{for all e} \qquad (5)$$

$$\sum_{d} \sum_{s \in S_{ed}} x_{esd} \leq 6 \quad \text{for all e} \qquad (6)$$

Note that conditions 7), 8) and 9) are not modeled in the preceding constraints. Modeling them is straightforward but tedius, requiring the use of additional notation without adding to the basic understanding of the requirements conveyed by their verbal description.

The Objective Function:

A perfect schedule (as distinguished from an optimal schedule) occurs when the number of employees assigned, by quarter-hour periods throughout the week, equals the number wanted, and no employee works less than his or her minimum desired number of hours during the week, while respecting the various constraints specified. Given the non-homogeneous character of employees, the fluctuating requirements, and the attributes of the various shift types, a perfect schedule is frequently impossible.

An optimal schedule is one that minimizes some weighted function of the shortages and overages, by quarter-hour periods throughout the week, plus a weighted function of each employee's hours below the desired minimum.

We define:

WU = a weight to penalize falling below forecasted requirements (yielding a shortage)

WV = a weight to penalize exceeding the forecasted requirements (yielding an overage)

W_e = a weight to penalize falling below the desired (as opposed to required hours for employee **e** to work during the week)

We seek, therefore, to:

$$\text{Minimize} \quad WU \sum_{p,d} u_{pd} + WV \sum_{p,d} v_{pd} + \sum_{e} W_e Y_e$$

The preceding formulation is not the only or most general our procedure can be made to handle, but represents the model applied to the 10 real world problems whose solution statistics we have reported. In this application, shortages were penalized more than overages by a ratio of WU to WV of roughly 4 to 1. Minimum desired employee hours were satisfied automatically by setting W_e to an internally computed parameter based on WU, WV and the user-specified seniority factor for employee **e.**

The number of variables in the integer programming formulation is of course independent of such parameter choices, and is potentially immense. If all possible placements of lunch and quarter-hour breaks were included in the schedules, the total number would amount to hundreds of millions of variables. (See the Appendix.) However, the admissibility of all such possibilities, or even uniformly structured subsets of all possibilities such as those resulting from the standard assumption of homogeneous

employees, would permit variables to be aggregated and thus effectively shrink their number to a minute fraction of those otherwise required. The simplifying homogeneity feature of [3] and [15], for example, makes it possible in these instances to deal with only about 300 to 500 variables in total.

In the setting of the general employee scheduling problem, where such implicit aggregation is not possible, we have created a component for our solution method that counts the total number of variables precisely for the real world examples we reported in Table II. These same counting rules were applied to data of the other papers to ensure that all reported and inferred numbers of variables were derived by the same means.

CONCLUSIONS

Given the best of today's state-of-the-art codes for solving very large scale integer programming problems, the general employee scheduling problem clearly does not lend itself to practical solution by standard procedures.

It is encouraging both in view of the practical significance of the general problem, and its combinatorial complexity, that a method which combines elements of management science and artificial intelligence techniques can generate solutions of exceedingly high quality in very modest amounts of time. It is also noteworthy that such results have been achieved on a microcomputer.

The "leap" in size by comparison to previous comparable scheduling applications reported in the literature demonstrates that the effective solution of truly large-scale scheduling applications is possible, contrary to widespread belief. It is inviting to believe that similar gains may be possible for other combinatorial zero-one applications, perhaps by similar strategies.

REFERENCES

1. Baker, K. R. and Magazine, M. J., "Workforce Scheduling with Cyclic Demands and Days-Off Constraints," MANAGEMENT SCIENCE, Vol. 24 (1977), pp. 161-167.

2. Brownell, W. S. and Lawerre, J. M., "Scheduling of Workforce Required in Continuous Operations Under Alternative Labor Policies," MANAGEMENT SCIENCE, Vol. 22 (1976), pp. 597-605.

3. Buffa, E. S., Cosgrove, M. J., and Luce, B. J., "An Integrated Work Shift Scheduling System," DECISION SCIENCES, Vol. 7 (1976), pp. 620-630.

4. Crowder, H., Johnson, E. and Padberg, M., "Solving Large Scale O-I Linear Programming Problems," OPERATIONS RESEARCH, Vol. 31, No. 5 (1983), pp. 803-834.

5. Glover, F., "Heuristics for Integer Programming Using Surrogate Constraints," DECISION SCIENCES, Vol. 8, No. 1 (1977), pp. 156-166.

6. _____, "Future Paths for Integer Programming and Links to Artificial Intelligence," CAAI Report 85-8, Center for Applied Artificial Intelligence, University of Colorado, Boulder, October 1985. To appear in Computers and Operations Research, P. Ghandforoush, ed.

7. _____, McMillan, C. and Novick, B., "Interactive Decision Software and Computer Graphics for Architectural and Space Planning," CAAI Report 85-3, Center for Applied Artificial Intelligence, University of Colorado, Boulder, March 1985. To appear in Annals of Operations Research: Algorithms and Software for Optimization, Vol. 4.

8. Henderson, W. B., and Berry, W. L., "Heuristic Methods for Telephone Operator Shift Scheduling: An Experimental Analysis," MANAGEMENT SCIENCE, Vol. 22 (1976), pp. 1372-1380.

9. _____ and _____, "Determining Optimal Shift Schedules for Telephone Traffic Exchange Operators," DECISION SCIENCES, Vol. 8 (1977), pp. 37-41.

10. Keith, E. G., "Operator Scheduling," AIEE Transactions, Vol. 11, No. 1 (1979), pp. 37-41.

11. Lowerre, J. M., "Work Stretch Properties for the Scheduling of Continuous Operations Under Alternative Labor Policies," MANAGEMENT SCIENCE, Vol. 23 (1977), pp. 963-971.

12. Mabert, V. A., and Raedels, A., "The Detail Scheduling of a Part-Time Work Force: A Case Study of Teller Staffing," DECISION SCIENCES, Vol. 8 (1977), pp. 109-120.

13. Morris, J. G., and Showalter, M. J., "Simple Approaches to Shift, Days-Off and Tour Scheduling Problems," MANAGEMENT SCIENCE, Vol. 29 (1983), pp. 942-950.

14. Segal, M., "The Operator-Scheduling Problem: A Network-Flow Approach," OPERATIONS RESEARCH, Vol. 22 (1974), pp. 808-823.

APPENDIX

Shift Definitions Used in the Runs Cited

Shifts are from 2 to 7.5 hours in length (not counting lunch) as follows:

1) Shifts longer than 5 hours have one lunch;

2) Shifts from 2 to 5 hours have no lunch;

3) All work sessions are at least 1.5 hours in length;

4) Fifteen minute breaks are not scheduled.

In runs cited, schedules were created covering 76 periods each day Sunday through Thursday and 80 periods each day Friday and Saturday.

This shift description results in 158 different shifts that can be assigned to begin at any period. Over 76 periods and with 30-minute breaks, 7,914 different shifts could be considered for an employee with unlimited availability (the number is less than 76 * 158 because no shifts can begin in the last 7 periods, only one shift can begin in the eighth period from the end, and so forth). For the entire week 56,662 different shifts are theoretically possible for such an employee, yielding a potential of 5,666,200 shifts for 100 employees.

The non-homogeneity of employees, which restricted available work days and available periods within days differently for each, caused the actual number of shift alternatives to be less than this quantity, though the total number of employees sometimes exceeded 100. The counters placed in our program indicated the actual number of shifts ranged from a low of 1,158,117 to a high of 4,999,580 shifts, as noted in Table II.

In other applications where it is desired to schedule 15-minute breaks, the number of variables involved soars. A typical 7.5 hour shift with a lunch and two 15-minute breaks has 375 variations compared with 19 variations for the 7.5 hour shift used in the above restaurant runs. We are currently experimenting with another version of our procedure which is handling over 26 million IP variables.

ISIS PROJECT IN REVIEW

Dr. Petros N. Papas
Manager, Computer Integration and Software Systems
Westinghouse Advanced Production Technology Systems Group
Pittsburgh, Pennsylvania

INTRODUCTION

This paper covers the History, Achievements, Lessons Learned, Present Status, and Future Plans of the ISIS Project. It is not a technical treatise of the code. ISIS stands for Intelligent Scheduling and Information System. It has been a joint development effort of Carnegie Mellon University, Westinghouse Electric Corporation and the U.S. Air Force.

The scheduling problem is most complex. The ISIS project is a beginning towards solving it.

ISIS aims at scheduling a job shop through constraint-directed reasoning. It uses a knowledge representation language to model the production environment and uses AI and heuristic search techniques to construct the schedule.

HISTORY

Carnegie Mellon University established the Robotics Institute approximately six years ago and within it the Intelligent Systems Lab headed by Dr. Mark Fox. Westinghouse became one of the original and strong supporters of the Robotics Institute with a multi-year funding commitment and plans for cooperative research. As a result, several joint research projects were initiated, including ISIS. Projects were chosen, their funding established, and their progress reviewed by a Westinghouse high-level committee.

The idea for the ISIS project came from a need at the Westinghouse Turbine Components Plant at Winston-Salem, North Carolina. The

need was to increase the throughput of orders while maintaining efficient utilization of all machines.

Let us digress for a moment and look at the scheduling problem.

The development of schedules to govern the production of orders in a job shop is very complex. They tell me that just ten orders through just five operations results in ten factorial to the fifth power possible combinations or schedules. The situation within an actual manufacturing facility is much more difficult. The number of orders, operations and resources is substantially greater. Also, the dynamic nature of the shop further complicates the problem. I am referring to machine breakdowns, order changes, engineering changes, tool breakdowns, tool unavailability, operator absenteeism, reworks, etc.

The major initial conclusion from the CMU study is that scheduling is a constraint-directed process. Westinghouse schedulers at Winston-Salem spend 80% of their time determining constraints and their impact. The objective of scheduling, after all, is not only to meet the due dates but to construct schedules which satisfy all or most of the constraints. Five catagories of constraints have been identified by the CMU research:

- Organizational Goals: Customer requirements, work-in-process, objectives.
- Physical Limitations: Machine capabilities, tooling capabilities, and product dimensions.
- Causal Restrictions: Operation precedence, resource requirements for each operation.
- Availability: Availability of resource to perform operations.
- Preference: Qualitative preferences for operations, machines or resources. The rules-of-thumb, that each scheduler uses to schedule a plant.

Also, ISIS schedules in four levels:

- o Lot Selection
- o Capacity-Based Scheduling
- o Detailed Scheduling
- o Reservation Selection

The core of this software is the beam search technique used in detail scheduling.

The first phases of the research were spent at defining the nature of the problem and also as a learning time for the CMU team. This team went beyond just familiarizing themselves with the problem. In the absence of a designated Westinghouse domain expert, they became the domain experts--the scheduling experts. This initial progress was slow, as expected.

After a couple of years (around 1982) the Westinghouse Review Committee decided that other projects ranked higher in priority and funds were shifted away from ISIS. As a result of the reduction of funds from Westinghouse, Mark Fox solicited and received support from the U.S. Air Force and CMU to continue the AI research in this most important manufacturing area.

I became involved in ISIS in the beginning of 1983 as part of a renewed interest by the Westinghouse Corporate Productivity & Quality Center. We might say that Westinghouse again recognized the fundamental nature and importance of this research.

We established a structured and focused effort aimed at a demonstration of the concept. A Westinghouse domain expert from Winston-Salem was set as the project leader. The demo deadline was December of 1984. The objectives of the demo were clearly spelled out well ahead of time so that all team members (Westinghouse and CMU) knew what was expected.

We also established a Westinghouse ISIS Design Review Committee made up of key manufacturing people. It is amazing what deadlines can do. The team went full speed ahead and not only met but exceeded all objectives including delivery by November 1984. The demo was a resounding success, from all viewpoints. It was shown to Westinghouse upper management, various Westinghouse Corporate and other committees and also the U.S. Air Force and CMU.

The next logical step was to take ISIS into production. Here is where we ran into a snag. In order to fully understand the next events, I have to digress for a moment and describe to you the Westinghouse Winston-Salem plant. It is called the Turbine Components Plant and it manufactures, among other things, turbine blades.

The plant possesses forging, fabrication, machining and assembly operations. The main characteristic of the plant is that of high-precision machined parts for the Electric Utility Industry. The plant also has a large number of DNC tools and the first robotic cell for swaging of steel bars, also completed as a joint Westinghouse/CMU project. Incidentally, the result of that research is a high level, AI technology-based language called CML (Cell Management Language) that Westinghouse is now marketing and using.

The plant routinely manufactures 10,000 different types of parts and fills approximately 7,000 customer orders per year, each order requiring about 20 machining or manual operations, including precision measurements. It ships a quarter of a million parts per year. ISIS was developed with the tapered blade section of the shop in mind. This section represents approximately 1/3 of the machining capacity of the plant.

The Winston-Salem plant has always been progressive and open to trying new ideas and was ideal for these projects.

With this as the background, it can now be seen that the demonstration took place at Winston-Salem in a live, real environment.

This next step, that of putting the system in production at Winston-Salem, did not take place. What was needed is a more robust version of ISIS, new computer hardware, and building of the interfaces to other systems. In other words, we had to translate ISIS from SRL to "industrial strength" software. At that point, the decision was made to discontinue work at Winston-Salem and instead opt for a smaller version of ISIS that can be made into a production package using commercially available hardware and software. This is now taking place. Whenever this new version is expanded to include all the features needed at Winston-Salem, then ISIS will be installed at the plant.

The latest version is called ISIS-II and has been shown at the last IJCAI Conference in August at UCLA. Those of you who were there might have seen it.

Let me hasten to add that this sequence of events is not a surprise but a natural progression from a breadboard prototype to a production model, just as it happens in the development of any product.

ACHIEVEMENTS

So much for history. Now let us look at the achievements.

1. Research has begun in this most important area of manufacturing. As I mentioned earlier, I consider scheduling a most intellectually challenging and important endeavor.

2. We have shown that this type of problem can be tackled by the application of AI/Expert Systems technology.

3. ISIS has contributed (along with many other efforts) to the creation of interest among academic circles for manufacturing problems. In other words, practical manufacturing problems are now perceived as not only important to solve but also interesting and intellectually challenging.

4. A Ph.D. thesis has been published on this subject: Dr. Mark Fox's thesis on Constraint-Directed Reasoning.

5. Five categories of constraints were recognized.

6. A wealth of research has come forward from the ISIS/CMU in this area. This reasearch is opening a whole new spectrum of ideas and products. One of the new products is a manufacturing scheduling algorithum that was announced by the Carnegie Group, Inc. of Pittsburgh.

7. The demonstration of the concept. This was a major accomplishment by the combined efforts of the Westinghouse and CMU teams. The demonstration included both the interactive and automatic methods of scheduling as well as a limited "what-if" simulation capability.

LESSONS LEARNED

Let us now focus on the lessons learned from this research effort. Most of these lessons center around the key elements of transferring the technology from the University to industry and then the user.

The most obvious lesson relearned was that the University is best suited for research and industry is best suited in developing and packaging products. It is something we knew before, but those of us who don't learn from history are doomed to repeat it.

We had unmatched expectations. The University says "give me the money, I'll give you research--no guarantee on results." Industry says, "I expect a list of milestones and deliverables for this number of dollars." Obviously not a congruent set of perceptions and expectations. The relationship has to be thought out so it can benefit both parties, as it did in this case.

Next lesson learned was the fact that we must choose a stable environment and robust, industrial strength software as the vehicle for development of a production system. We started out with SRL (Schema Representation Language) from the University which was adequate for research but totally inadequate for production.

SRL was easy to use. It had all kinds of handlers, subroutines and productivity aids that improved our capacity to do research and also to quickly prototype and develop a system. But SRL was never really tested in a production environment and only handled smaller problems.

The third lesson learned is that it is best to develop your own in-house AI capability or use a commercial AI firm for the development of an actual product. University expertise can best be utilized on an as-needed consulting basis during product development. University is best for research. Since the AI resources are limited in this country, it is a shame to waste scarce university research talent in writing production code.

Another lesson I would liked to have learned before starting out on ISIS, is how to pick as simple a problem as possible to get started. We didn't know any better; that is why we picked a difficult problem to tackle. Right now we have scaled back to tackling a problem of a more manageable size. I believe we will have results by the end of the year.

The last lesson learned is that the cost is going to be higher than you think and the results less than you expected. Be realistic in your planning. Don't be dissappointed. Give your team time to develop ideas and a chance to come through.

PRESENT STATUS

What is happening today? For once we are developing a scaled back version called ISIS-II. What this means is as follows. The demonstration project covered both the scheduling of the machines

and also generated a routine, that is a sequence of operations. So, ISIS was both a scheduler and generative process planner. In ISIS-II we have chosen to concentrate on the scheduler, assuming that the routines are given. ISIS-II also concentrates on the interfaces to traditional MRP systems. In this way, we can focus our attention entirely to the solution of the scheduling problem. The attached figure shows the ISIS-II Architecture.

This first third of this year was spent in the selection of hardware and software. We put the basic beam-search algorithm in various combinations of hardware and software in order to decide on the best combination. We chose KEE from IntelliCorp and EXPLORERs from Texas Instruments. ISIS-II is now being developed and should be available for pilot testing by year-end. We are now selecting several pilot sites within Westinghouse in order to test out the code.

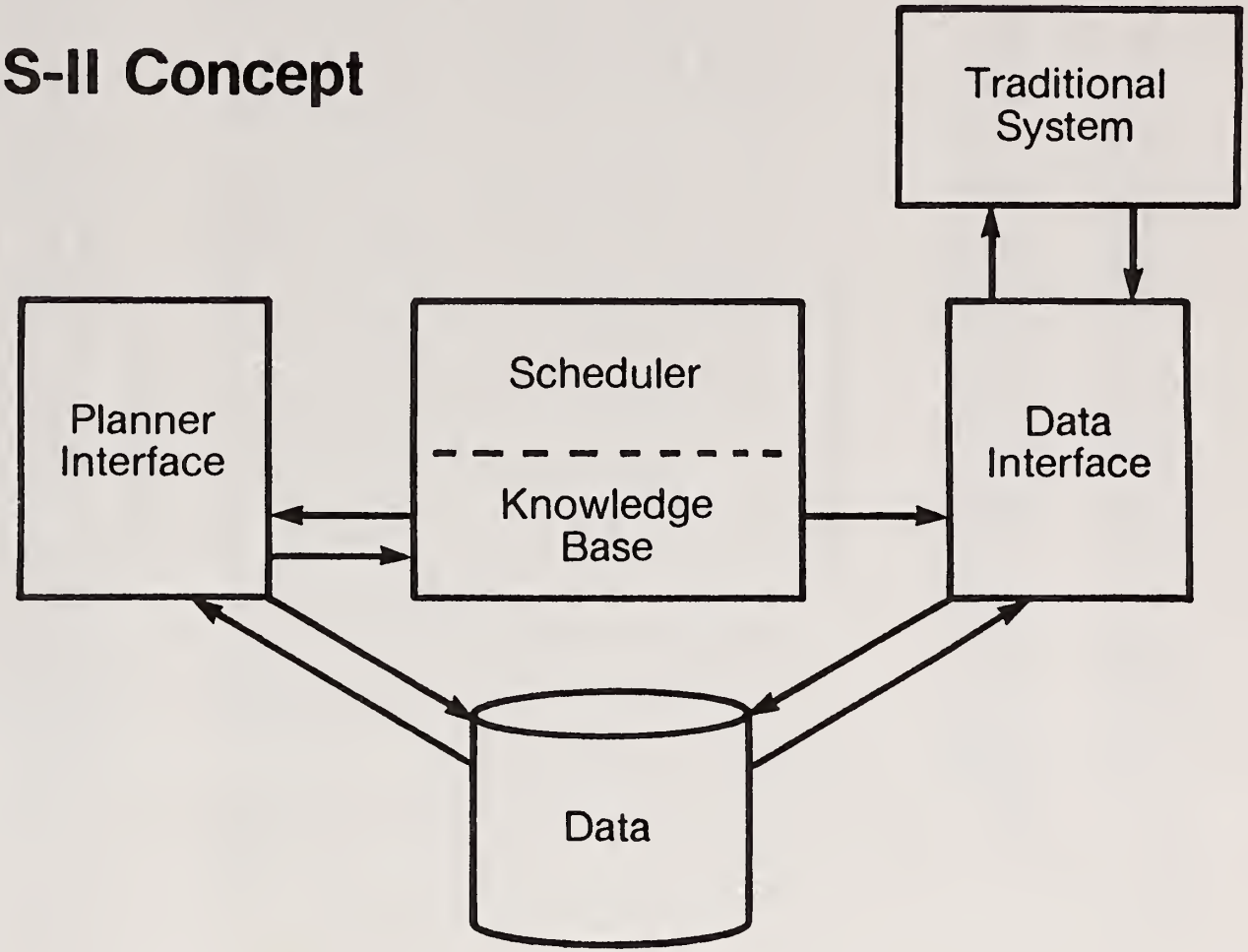
FUTURE PLANS

Obvious future plans are of two types: The expansion of the use of ISIS-II to as many locations of Westinghouse as deem it applicable and the development of enhancements. Possible enhancements of ISIS-II are:

- Handle larger manufacturing plants
- Handle several hierarchies of schedules
- Generate process plans
- Interface to tooling systems

After all this is accomplished, I am sure that we will find other superb manufacturing opportunities to tackle.

ISIS-II Concept



DYNAMIC CONTROL IN AUTOMATED MANUFACTURING: A KNOWLEDGE INTEGRATED APPROACH

James G. Maley, Sergio Ruiz-Mier, James J. Solberg
Center for Intelligent Manufacturing Systems
Purdue University, West Lafayette, IN 47907

INTRODUCTION

Flexibly automated facilities can process a wide variety of products under different constraints and objectives. They thus require manufacturing control strategies capable of facing an environment of continual change. To control and schedule within such an environment, a system which can be responsive to product and process requirements, machine breakdowns and delays, engineering changes, and improvement opportunities, is needed. Such a control system does not fall into the realm of any current manufacturing solution techniques. Dynamic operation of a production system in this environment requires something more than exact optimization, heuristic algorithms or stochastic estimates. Recently, an effort has been made towards increasing the flexibility of the controlling process itself to handle unanticipated problems and situations [see Nof, et al. 1980 and Solberg, et al. 1985]. Still, according to Naylor [1985] and in the authors' opinions, the amount of flexibility required to adapt to the varying situations facing manufacturing has not yet been reached. The work contained herein goes beyond some of the concepts from Nof et al., Solberg et al., and Naylor, such as a modular scheduling and control framework and a generic implementation to face differing factory situations and assumptions, to address the problem of controlling the flow of parts and information. As an example of the described strategy, we present an application to automated guided vehicles operating in a flexible environment.

BACKGROUND

The term Flexible Manufacturing Systems (FMS) has been a popular way of describing automated manufacturing cells which contain an element of flexibility in their manufacturing of goods. An excellent overview of this stage of the development of automated facilities is found in Dupont-Gatelmand [1982]. Dupont-Gatelmand discusses the history, definition and implementation of flexible automation up through the early eighties. Her work provides an excellent introduction to the area with numerous examples of working FMS's. Stecke [1984] provides a more recent description of FMS's in terms of the problems encountered with system design and implementation. Stecke's work was published during the beginning of the trend away from what was defined as FMS's to the more wholistic view of factory automation - Computer Integrated Manufacturing or CIM. According to Stecke's classification, the research in this project is in an effort to solve the combined scheduling and control problems in an automated manufacturing facility.

For an overview of the production scheduling literature in general, see Graves [1981]. Two papers by Iwata, Murotsu, Yasuda and Oba [Iwata et al. 1982 and Murotsu et al. 1983] offer a unique scheduling strategy specific to FMS's. Both papers by Iwata et. al. base their scheduling and control capabilities on the heuristic strategy of a "pull-system". Briefly the pull-system scheduling strategy is based upon the idea that the part is moved according to the status of the manufacturing system at the time of or close to the time of its impending movement. Thus instead of being planned ahead of time and then "pushed" through the system, the part is "pulled" through as time progresses. Some earlier work completed by Lewis et al. [1980] recommends this type of automated facility control. Further, Lewis et al. go on to show that a data flow approach to controlling computerized manufacturing systems provides a method of meeting the need for increased flexibility. Some of the ideas of pull system are introduced in the specific example of our procedure.

The solution strategy which we present herein is developed for an automated manufacturing process, yet our focus in the example is on the material handling subsystem. The automated material transport systems, which are widely used in flexible systems today, are automatic guided vehicles. Maxwell and Muckstadt [1982] present the ideas behind designing a guided vehicle system for a facility and then go on to implement their recommendations on a simple factory model. Egbelu and Tanchoco [1984] provide an overview of guided vehicle systems and characterize the differing methods of implementing scheduling and routing rules for those systems. The type of AGV system used in this project can be characterized as a uni-directional vehicle operating in aisles which have room for vehicles to meet and pass by each other. A paper by Parodi [1984] provides a route planning system based upon the "pull" system principle, described above. In the paper an autonomous vehicle (possibly an automated guided vehicle) is routed in a dynamic and varying environment. This environment is similar to that faced by material transport systems on the factory floor. In our example we implement a modified version of Parodi's algorithm for routing the guided vehicles.

The idea of combining simulation and optimization into a tool that will assist the problem solver is not new. In fact, the use of this combination in computer integrated manufacturing has been recently looked into by Chu [1984]. Chu's research had the combined goal of developing a system which would provide adaptive control of a real system while simultaneously modifying a model of the system for a dynamic manufacturing environment. Chu demonstrated the feasibility of his ideas on a single robot for material handling. Fisher and Thompson [1963] describe a learning system that develops job-shop scheduling rules. Fisher and Thompson use parameter variation to guide combinations of scheduling rules to improve the performance over any of the rules used individually. Some of the self-improving or adapting ideas utilized in this paper are taken from the Fisher and Thompson paper. The combination of both Chu's work and Fisher and Thompson's work has been made possible through a knowledge based environment developed by Ruiz-Mier and Talavage [1985].

THE DYNAMIC SOLUTION STRATEGY

The concept of real-time operation of a controlling element of a manufacturing facility conveys various connotations to different people. From our viewpoint, real-time operation is an ideal to strive for; however, it may be hard to achieve.

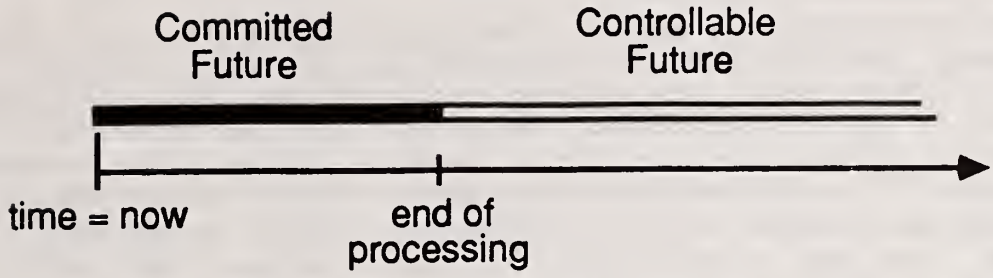


Figure 1

In discrete parts manufacturing, the exertion of control over processes does not necessarily cause an immediate effect on the system. This is due to the fact that many processes, once started, are committed and must proceed until completion. For controlling purposes, we can visualize time (see figure 1) as being divided into a fixed period, where decisions cannot affect the system, and a flexible or controllable future. With this in mind, let us look at the proposed solution strategy.

We identify five key components in the operation of a scheduling/control process which must be intimately coordinated. These are: real-time feedback from the operation of the facility, guidance from a historical knowledge base, forecasting of what is to come, direction of goals and constraints from management and world data, and improvement of the process from a planning module. Shown in figure 2 is a diagram of the interrelations required.

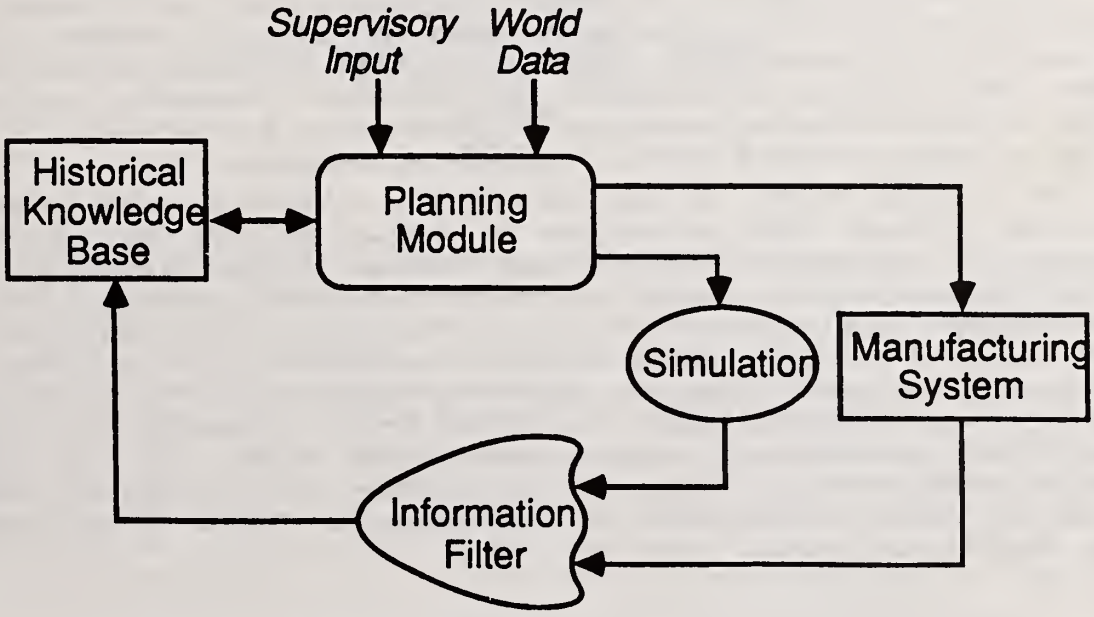


Figure 2

The closed loop system provides not only a visual representation of the

interrelationships but also a guide for actual implementation by means of an object-oriented programming paradigm.

The information filter uses the prevailing constraints, the present objectives, and the current state of the facility to discern relevant facts about the system. These facts are necessary to maintain the historical knowledge base, direct the planning and to evaluate the system's performance. Based upon the goals (objectives), the type of information needed from the filter is implicitly defined. The information filter determines which actual data flowing through it is relevant and which is not.

Real-time feedback from the manufacturing facility is necessary to determine precisely how the current scheduling/control decisions are performing. This information provides the basis from which the planning module can modify its operation to improve the decision making process in force. Also the continuous feedback of information provides data concerning the status of the facility - in particular the current system constraints. This data can be used by the planning module to take advantage of specific situations as they arise.

In general, we can say that the problem of optimizing the system can be characterized by a systematic search for a solution. The general idea behind using the historical knowledge base is to guide the search or at least provide starting points for the search and to avoid reproducing previous work. An important factor in the development of the historical knowledge base is the determination of the amount of data stored and the way the data is stored.

The ability to see the future would provide the opportunity to do processing off-line. However, with the dynamic nature of the facility emphasized above, this luxury is not available. Our method of forecasting the future includes the added possibility of testing scheduling/control decisions prior to their implementation. As can be seen in figure 2, we propose a simulation of the facility to give us a way of testing the current plan with what might happen in the future.

The direction of goals from management represents another way of maintaining an actual representation of the operation of the system. One of the dynamic aspects of the manufacturing system is the changing set of goals which management wishes to achieve. By permitting the infusion of this supervisory information, the system is always working towards the current goal set. This goal set is constrained by the status of both the system itself and the world data affecting the system.

The planning module is the heart of this scheduling/control process. It determines precisely which decisions are to be made and how those decisions are implemented. It chooses the search strategy to achieve the goals given to it. The key to the planning module's unique role in this closed loop information flow is it's interaction with the historical knowledge base. Chu [1984] tries to optimize this type of planning routine by storing various scenarios in a historical data base. Then using a pattern matching scheme, his system chooses the best solution to the current problem based upon the historical problems. We see the historical knowledge base interaction as providing a starting point to continue optimizing the planning process as opposed to being the solution selector. Thus, not only does the planning routine schedule and control the facility, but the manner in which it operates changes through time based upon historical information.

EXAMPLE: A KNOWLEDGE INTEGRATED APPROACH

Briefly, this example entailed implementing a scheduling/routing procedure within a simulated manufacturing environment. This example is not a complete implementation of the proposed formalism; however, it illustrates the formalism's

underlying concepts and gives promising results. In this automated manufacturing environment, automated guided vehicles provide the means of material transport. Each vehicle uses its own conceptualization of the facility layout to maintain a history of actions and to implement a demand-pull manufacturing control scheme. The actual scheduling/routing of the parts through the factory used the A* search algorithm [Hart et al. 1968] to determine the weighted shortest path between the points which it had to travel.

The automated facility used to demonstrate the feasibility of the dynamic optimization strategy is a modified version of an FMS as described in Dupont-Gatelmand [1982], figure 3.

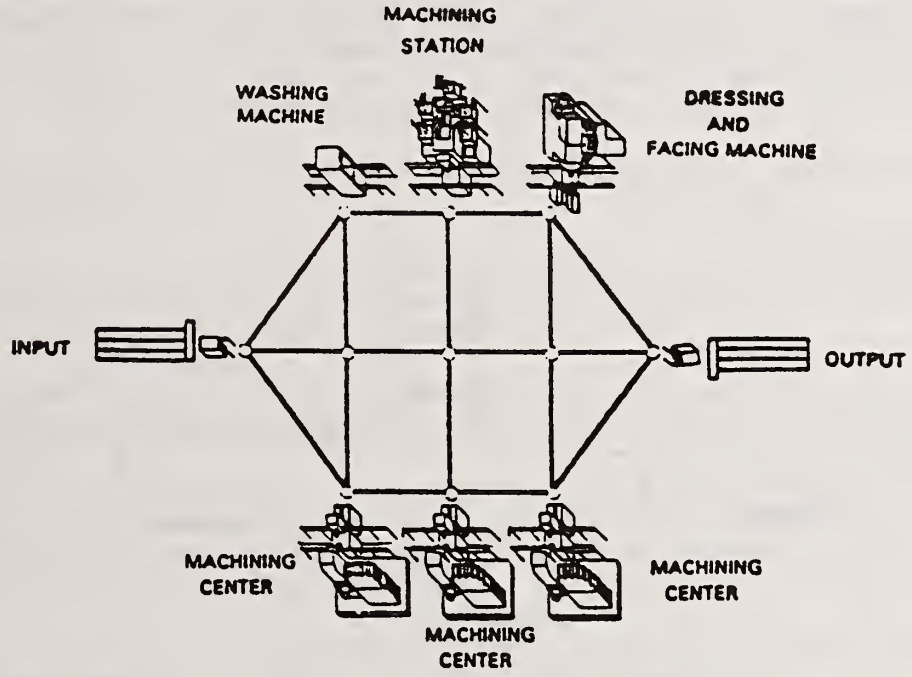


Figure 3

As one can see from the figure, a network description of the facility has been created. The network is represented using a frame-based paradigm. Given this representation, the problem modeled herein is to have the AGVs traverse the paths by a simple set of rules (use the AGV with the highest amount of idle time or the first available AGV) and move as many parts as possible through their sequence of processes. The goal of this process is to have each vehicle's representation of the facility layout develop to such a point that each vehicle has its own through-ways. These through-ways will give the vehicle ample opportunity to travel the quickest way possible without delays (ie: blocking). Since the nature of the manufacturing environment is quite dynamic, a steady-state map for each vehicle is not a proper goal. However, the ability of the vehicles to adapt continuously over time provides the means for keeping up with the changes.

Referring to figure 2, we now show how this example fits into our solution strategy. The desired objective function, in this case to maximize throughput, is specified as supervisory input to the system. The manufacturing system shown in figure 3 is simulated in the CAYENE environment [Ruiz-Mier and Talavage 1985] and

performs as described above. Data from the simulation is passed through the information filter. In this example, the filter scans the vehicle's performance in the simulation and correlates it with the planning module map. This information concerning the vehicle's view of the facility is stored in the historical knowledge base. Each iteration of the procedure adds a new set of frames to the historical knowledge base from the information filter. Finally, the planning routine compares the performance measures stored in the historical knowledge base and updates the individual AGV maps by modifying the weights on the paths to improve the system objectives (ie: to maximize throughput subject to the current constraints).

The results of a number of iterations of this procedure for a given configuration of the manufacturing system are shown in figure 4. Iteration 0 depicts the system as unit weights on the inter-node paths. Iterations 1 through 3 show how the system improves its throughput performance with adjustments to the individual AGV's view of the paths. Note that after only four iterations a 27% improvement in the throughput and a 10% improvement in the time in the system is achieved.

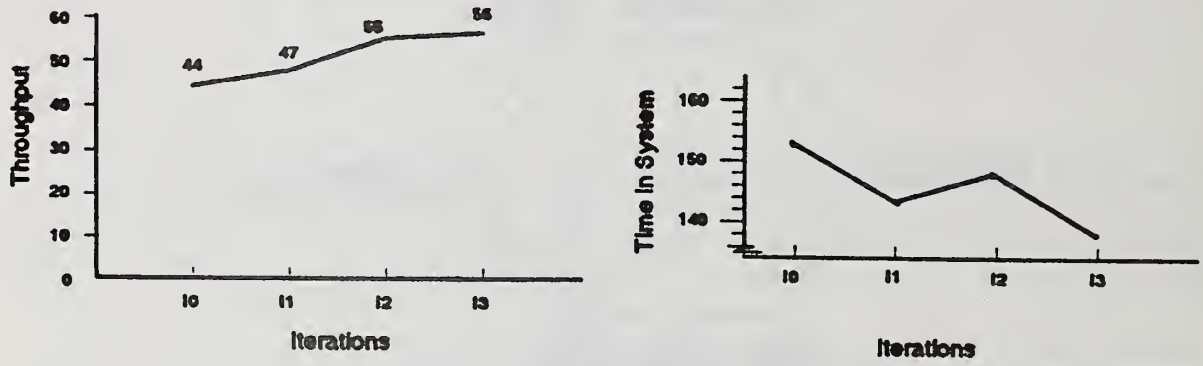


Figure 4

CONCLUSIONS

From the example demonstration of the dynamic formalism presented in this paper, we can conclude that the proposed system is responsive to the environment. The marked improvement in throughput in this example shows that the formalism is effective. The ability of the dynamic solution system to improve over time, as demonstrated, provides the impetus for further work into the completion of the entire proposed system. The key to a full scale implementation will be the development of the system in a language suited for faster computation time.

BIBLIOGRAPHY

Chu, Chi-Chung, "An Adaptive Decision Making Methodology for Material Handling Equipment in a Computer Integrated Manufacturing System", Ph.D. Dissertation, School of Industrial Engineering, Purdue University, West Lafayette, Indiana, December 1984.

Dupont-Gatelmand, Catherine, "A Survey of Flexible Manufacturing Systems", *Journal of Manufacturing Systems,* Vol. 1, No. 1, pp. 1-16, 1982.

Egbelu, Pius J. and Jose M. A. Tanchoco, "Characterization of Automatic Guided Vehicle Dispatching Rules", *International Journal of Production Research,* Vol. 22, No. 3, pp. 359-374, 1984.

Fisher, H. and G. L. Thompson, "Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules", pp. 225-251 in *Industrial Scheduling,* (Muth, John F. and Gerald L. Thompson, editors), Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1963.

Graves, Stephen C., "A Review of Production Scheduling", *Operations Research,* Vol. 29, pp. 646-675, 1981.

Hart, Peter E., Nils J. Nilsson, and Bertram Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", IEEE Transactions of Systems Science and Cybernetics, Vol. SSC-4, No. 2, pp. 100-107, July 1968.

Iwata, K., A. Murotsu, F. Oba, and K. Yasuda, "Production Scheduling of Flexible Manufacturing Systems", *CIRP Annals 1982,* Vol. 31, No. 1, pp. 319-322, 1982.

Lewis, W.C., M.M. Barash, and J.J. Solberg, "The Optimal Planning of Computerized Manufacturing Systems: Data Flow CMS Control System Architecture", Report No. 18, NSF Grant No. APR74-15256, School of Industrial Engineering, Purdue University, West Lafayette, Indiana, December 1980.

Maxwell, W.L. and J.A. Muckstadt, "Design of Automatic Guided Vehicle Systems", *IIE Transactions,* Vol. 14, No. 2, pp. 114-124, June 1982.

Murotsu, Yoshidada, Fuminori Oba, Kazuaki Iwata, and Kazuhiko Yasuda, "A Production Scheduling System for Flexible Manufacturing Systems", pp. 957-971, in *Computer Applications in Production and Engineering,* (E.A. Warman, editor), North-Holland Publishing Company, 1983.

Naylor, Arch, "Integration, Flexibility, and Software", as presented at the NSF Workshop on System Integration Tools in Manufacturing, November 6-8, 1985.

Nof, Shimon Y., Andrew B. Whinston and William I. Bullers, "Control and Decision Support in Automatic Manufacturing Systems", *AIIE Transactions,* Vol. 12, No. 2, pp. 156-169, June 1980.

Parodi, Alexandre M., "A Route Planning System for an Autonomous Vehicle", *IEEE First Annual Conference on A.I. Applications,* pp. 51-56, 1984.

Ruiz-Mier, Sergio and Joseph Talavage, "Towards a Knowledge-Based Network Simulation Environment", *Winter Simulation Conference,* San Francisco, CA, 1985.

Solberg, James J., David C. Anderson, Moshe M. Barash, and Richard P. Paul, "Factories of the Future: Defining the Target", CIDMAC, Purdue University, NSF Grant Final Report MEA-8212074, January 1985.

Stecke, Kathryn E., "Design, Planning, Scheduling and Control Problems of Flexible Manufacturing Systems", *Flexible Manufacturing Systems - Operations Research Model and Applications, Proceedings of the First ORSA/TIMS Special Interest Conference,* pp. 1-7, Ann Arbor, Michigan, August 15-17, 1984.

A MANAGEMENT CONTROL APPROACH TO THE MANUFACTURING PLANNING AND SCHEDULING PROBLEM

Ronald F. McPherson, Graduate Research Assistant
K. Preston White, Associate Professor
University of Virginia
Department of Systems Engineering
Charlottesville, Virginia 22901

Introduction

Typical manufacturing facilities produce from hundreds to thousands of different products. Manufacturing planning and scheduling is the marshaling and coordination of these facilities in pursuit of the firm's goals. The effort requires the establishment and achievement of goals at all levels of the manufacturing organization. The efficiency of these functions impacts the economic health of a firm in a competitive environment.

The magnitude of the manufacturing planning and scheduling problem makes its representation, analysis, and optimization using a single model impractical in most situations. The data required to specify in detail the manufacture of thousands of products is overwhelming. In addition, the effort required to solve the problem usually increases exponentially with the number of parameters required to represent the problem.

One method for reducing the enormity of the problem, hierarchical systems, decomposes the overall manufacturing problem into a hierarchy of smaller, more manageable problems. The decomposition is characterized by parameter aggregation and problem time horizon. At upper levels of the hierarchy, problems address general requirements about large aggregates of products over long time horizons. Descending through the hierarchy, the problems involve more detail about smaller product aggregates over reduced time horizons. Problems at upper levels place constraints on the formulation and solution of problems at lower levels.

A number of hierarchical approaches have been proposed. Hax and Meal (1975), Bitran, Haas, and Hax (1981, 1982), Bitran and Hax (1977), and Hax and Golovin (1978) have presented hierarchical approaches for the batch-process and job-shop environments. A hierarchical MRP model has been developed by Anderson *et al.* (1981). Conditions for perfect aggregation and a general solution procedure for the imperfect aggregation case are presented by Axsater (1980).

Meal (1984) discusses the motivation and implications for management of the use of hierarchical models. The hierarchical structuring of the problem mirrors the organizational structure of most traditional manufacturing operations. Decisions made at the upper levels of the manufacturing organization involve long time horizons and aggregate parameters. The use of hierarchical planning and scheduling aids, which push detail to lower hierarchical levels, coincides with the preference of upper management to delegate detailed production decisions to lower levels of the organization.

Although the operations literature addresses the parameter representation and manipulation obstacles, the real-time dynamics of the manufacturing organization in establishing and pursuing goals in a changing

environment are seldom addressed (for an exception see Fox *et al.*, 1983). The hierarchical approaches proposed to date are for the most part static planning systems. These approaches typically do not deal with organizational dynamics associated with revising goals or achievement of plans. Abraham *et al.* (1985) discuss the limited usefulness of static models in real life situations. In the real world, manufacturing organizations operate in a dynamic environment. Planning models must be developed which produce robust goals that are valid under a range of production scenarios. Elements of the manufacturing hierarchy tasked with achieving plans must have a means for reacting to a dynamic problem space. All the factors which together compose the production scenarios form this decision problem space. The means provided for the control of production determine the ability of the organization to react to changes in the problem space. In response to changes in the decision problem space, two resource consuming chain reactions may be initiated within the organization; one which propagates down the hierarchy (associated with planning) and another which travels up the hierarchy (associated with goal achievement). The means for production control, while allowing the organization to react adequately to change, must limit these two reactions to prevent the organization from wasting effort. Current hierarchical methods do not address or provide a method for analyzing this problem.

The management planning and control literature holds promise for providing a means for analysis of the manufacturing organization dynamics resulting from the dynamic decision problem space. This literature addresses the control of organizations to derive and achieve goals in a dynamic operating environment. One proposed planning and management control framework relates the planning and goal achievement functions within the organization. This relationship is an important determinant in the effort expended by the organization in achieving the goals. Recasting the manufacturing planning and scheduling problem in a planning and management control format emphasizes the manufacturing organization dynamics and the roles of decentralization (decomposition), autonomy, and robustness on these dynamics.

The automated plant, although free of human decision-making, must cope with the same problem-scope and organizational-dynamics considerations that affect the traditional manufacturing organization. Problem-size-dependent processing time precludes single model formulations for automating the planning and scheduling decision process. Hierarchical approaches provide a method for creating a hierarchy of smaller decision models by decomposing the production scenario, based on time horizon and parameter aggregation. This decision model hierarchy must respond to changes in the decision problem space through efficient organizational dynamics. The management planning and control approaches, although derived for analysis of organizational dynamics where human decision makers dominate, can be applied to the automated decision model hierarchy.

In this paper, a manufacturing organization, composed of levels identified in the current operations literature, will be cast in the planning and management control format. In this format the scheduling level becomes primarily a goal achievement process. That is the scheduler is charged with the real-time achievement of a higher level planner's goals. Objectives for real-time scheduling decision aids which include scheduling autonomy needs will be developed based on the goal achievement process. The

first section summarizes the levels of the manufacturing hierarchy. Section two introduces the planning and management control framework used for analysis. In section three organizational dynamics of the manufacturing hierarchy, recast in the planning and management control framework, are discussed. The final section presents an analysis of the scheduling level in detail to develop objectives for real-time scheduling decisions aids.

Manufacturing Hierarchy

One possible hierarchical decomposition of the manufacturing planning and scheduling problem is shown in Figure 1. The upper four levels-- (1) manufacturing system planning, (2) production planning, (3) flow planning, and (4) scheduling correspond to those proposed by Abraham *et al.* (1985) and Maxwell, Muckstadt, Thomas, and van der Eecken (1983), and are similar to those developed by Hax and Meal (1975). The lower two levels, cell control and machine control, correspond to those proposed by Gerswhin *et al.* (1985). The organizational dynamics of production planning, flow planning, and scheduling will be explored within the planning and management control framework.

Manufacturing system planning determines the manufacturing resources needed to achieve the long term goals of the manufacturing organization. The parameters of concern are long run volumes and categories of products, labor, equipment, and information. Types and amounts of equipment, employment policies, and reporting requirements are typical of the decisions that are made at this level.

The decisions made at the manufacturing system planning level form constraints on decision making at the production planning level. Typical decisions at this level involve production rates for aggregate product classes which are feasible with respect to the market demands and capacity of the facilities and, with part and material reorder intervals.

Flow planning determines the production batch sizes for each product. The process steps and flow times for each product batch are set at this level. Again all decisions made at higher levels form constraints on these decisions.

The fourth level of the hierarchy is scheduling. This level is responsible for implementing the flow plan. The sequencing and coordination of products through the production facilities in a real-time manner which adheres to the flow plan is the main objective. Time horizons at this level are short.

The cell control level considers each machine in a process step and the resulting scheduling implications within this group. Machine control addresses the dynamics of the components of each machine within a process. These two levels are represented within the planning and management control framework by a single function labeled production processes.

The decisions made at each level are characterized by time horizon and aggregation of parameters. At the upper levels of the hierarchy, decisions are made which involve long time horizons and large groups of products. Stepping down the hierarchy more detailed decisions about smaller product

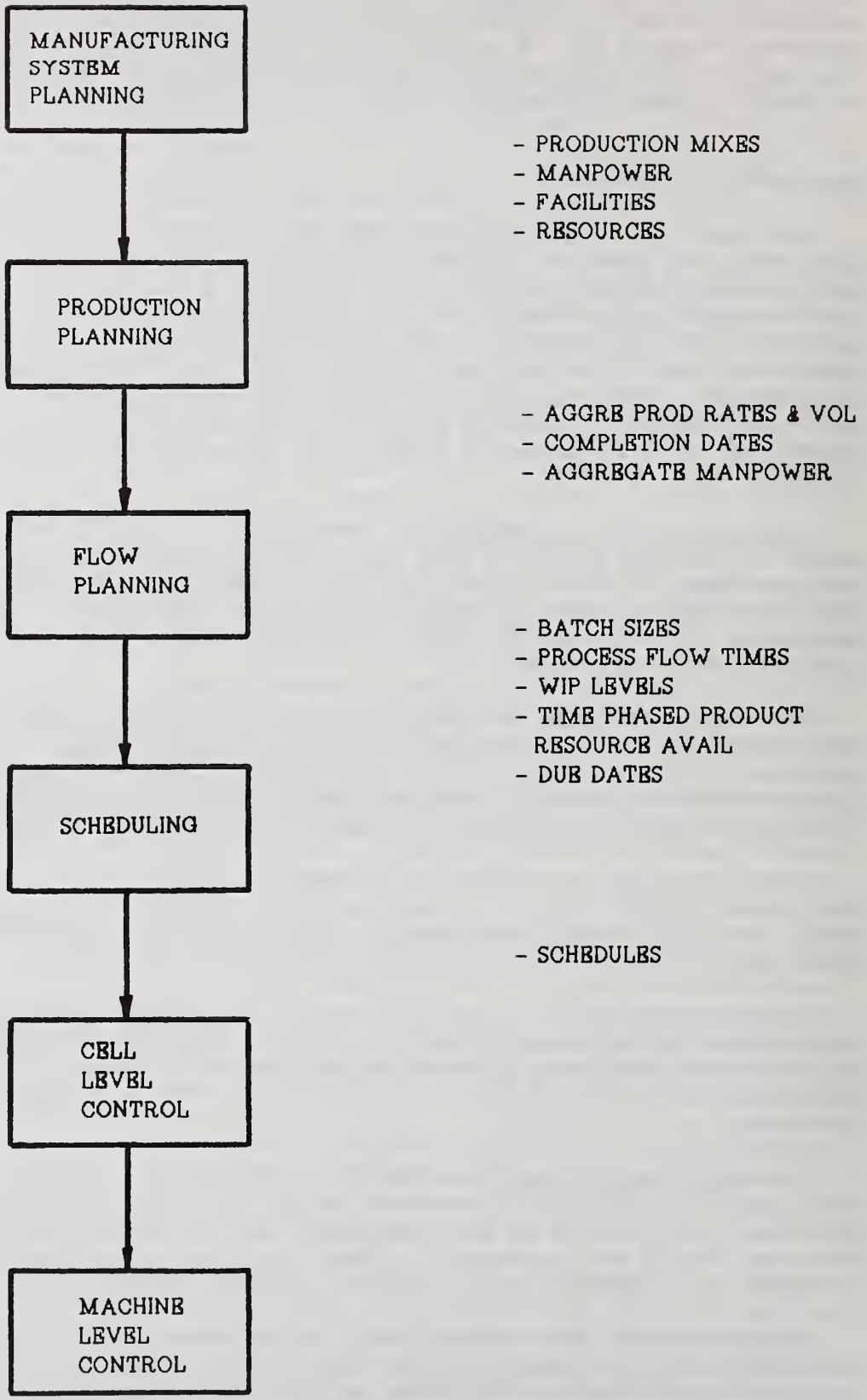


FIGURE 1 – A Hierarchical Decomposition of the Manufacturing Planning and Scheduling Problem

groups over shorter time horizons are made. This decomposition of the planning and scheduling problem is representative of the type found in the hierarchical manufacturing systems literature.

Planning and Management Control Framework

The planning and management control framework chosen for analysis of the organizational dynamics of the manufacturing hierarchy was developed by Middaugh and McPherson (1986). It was chosen for its rather generic hierarchical structure which places emphasis on decision processes for both goal determination and goal achievement. By modeling the general organization decision process as an open, purposeful, automatic control system, the authors capture the insights of previous frameworks such as Anthony (1965), Young (1966), Flamholtz *et al.* (1985), and Merchant (1985) within a single framework. Ackoff (1971) defines a purposeful system as one which can change it's goals or its structure or both in response to a stimulus. Use of the framework starts with a description of the manufacturing hierarchy in terms of the open, purposeful automatic control system model. This description associates the manufacturing decision activities with two generic functions: (1) Planning - activities pertaining to the establishment of goals and (2) Operations - activities to achieve the goals. Their placement within the model requires the manufacturing decision processes to possess certain characteristics if organizational control is to be maintained. Management control functions are generally defined as those activities which cause the operations functions to achieve their goals. A description of the organization decision process model and a sample of some of the issues that can be addressed follow.

The basic structure, shown in Figure 2, of the open purposeful automatic control system model consists of two optimization blocks which derive goals for a control loop tasked with achieving the goals. Goals are established through a two stage process of problem formulation and problem solution. Problem formulation includes not only determination of objective functions and constraints, but also the structuring of the control loop which contributes to the form of the constraints (*i.e.*, influencing how goals can be achieved). Although solution of the problem by optimization yields better goals, problem complexity usually forces satisficing to determine goals. Deviations from goals are used by the effector to determine commands to direct the activity in a manner which reduces the variances. The effector dynamics vary over time within the limits set by the optimization blocks. Generally, the effector chooses how to react to the discriminator signal. In a similar fashion the activity dynamics also vary over time, although some variations are due to sources outside control of the organization. Therefore, the activity can react differently to identical commands (depending on the dynamics in effect at the time).

The hierarchical nature of the model results from the complexity of the goal achievement functions. While planning functions are centered in the optimization blocks, the control loop is the location of the operation functions (*i.e.*, the control loop effects the goals of the planning blocks). In general, complexity of operational decision making requires the effector to possess planning functions (*i.e.*, the effector can formulate and solve inferior level problems as a means of generating commands to the activity).

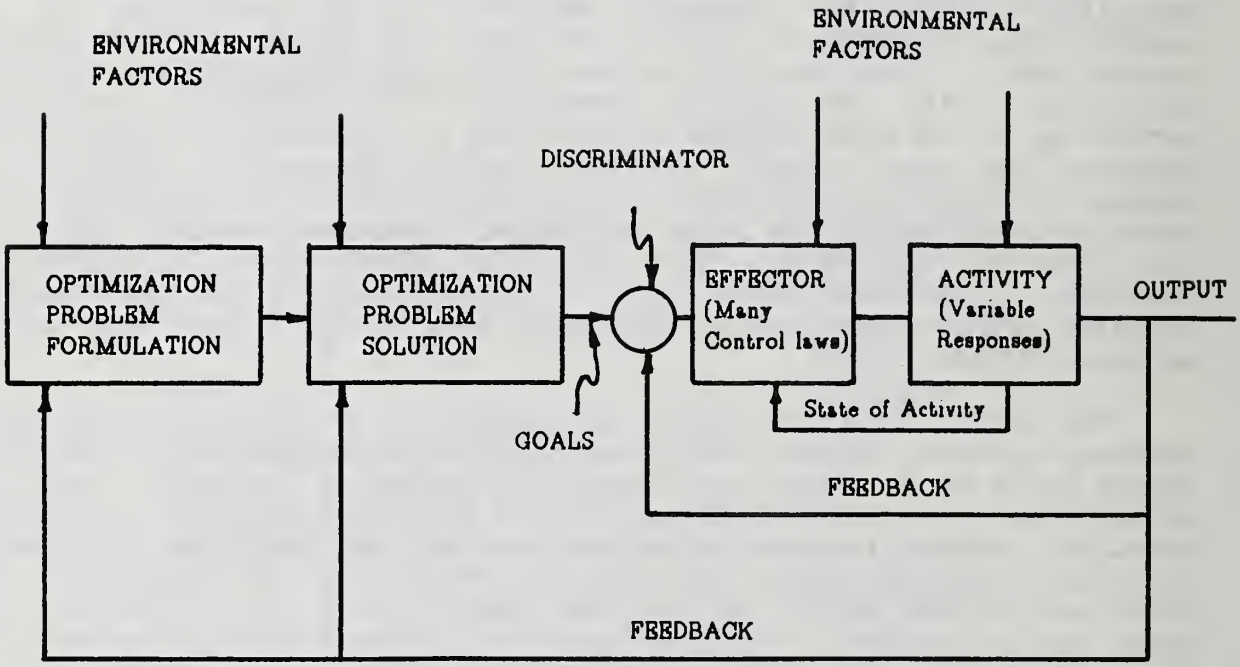


FIGURE 2 – Generic Organization Decision Process Model of Midduagh and McPherson

In this case the activity block, tasked with obtaining lower-level goals, takes the form of a lower-level control loop. This results in the hierarchical embedding of the basic model structure of Figure 2 in the nesting of the control loops. Figure 3 illustrates this generalization of the effector and activity blocks for two levels. A horizontal hierarchical structure is also possible as shown in Figure 4. In this example the planning blocks determine goals for two control loops at the same level. The actual structuring of the hierarchy is fitted to the scenario under analysis.

Issues in management planning and control include decentralization, autonomy, the selection of decision rules by the decision makers, and communication between decision makers. Decentralization and autonomy effect the positioning and scope of decision making authority within the hierarchy. These considerations, by providing the means for decision makers to react to changes in their problem spaces, effect the hierarchy's real-time achievement of goals in a dynamic environment. In a traditional organization, motivational methods, action constraints, and personnel controls are used to affect the selection of decision making rules. They increase the probability that decision makers select rules from the available sets at each hierarchical level which are not dysfunctional to the organization. In the automated plant, decision rule selection processes must also be designed. The management control methods can serve as a guide for approaching this problem as these address the more complex problem of control of human decision processes. The communications between parts of the hierarchy in the automated plant are not affected by motivational and behavioral considerations as these are in the traditional organization. In the remaining discussion, the paper assumes that sufficient methods are provided for proper selection of decision rules and communication of information. This is not an insignificant task and cannot be ignored in production management, but is beyond the scope of this paper.

Planning and Management Control Model of Manufacturing Planning and Scheduling

The manufacturing hierarchy presented earlier is naturally modeled by the planning and management control framework. Figure 5 illustrates how the production planning, flow planning, and scheduling levels are adapted within the model. The entire structure shown is the production planning level. Note that this is a control loop and as such is charged with achieving manufacturing system planning goals. The effector for this control loop is represented by the production planner and the activity by the two nested control loops containing the flow planning and scheduling levels. In this adaptation the two stage planning process described in the previous section has been combined into one block for clarity. Note how each inferior level control loop becomes the activity commanded by the adjacent superior level. Stepping out from the inner most control loop (scheduling level) the production processes are commanded by the scheduler (scheduling control loop effector). In turn the scheduling control loop is commanded by the flow planner (flow planning control loop effector) and finally the flow planning loop is the activity commanded by the production planner.

The same relationships defined previously for the manufacturing hierarchy are preserved in this model. The definition of the blocks and

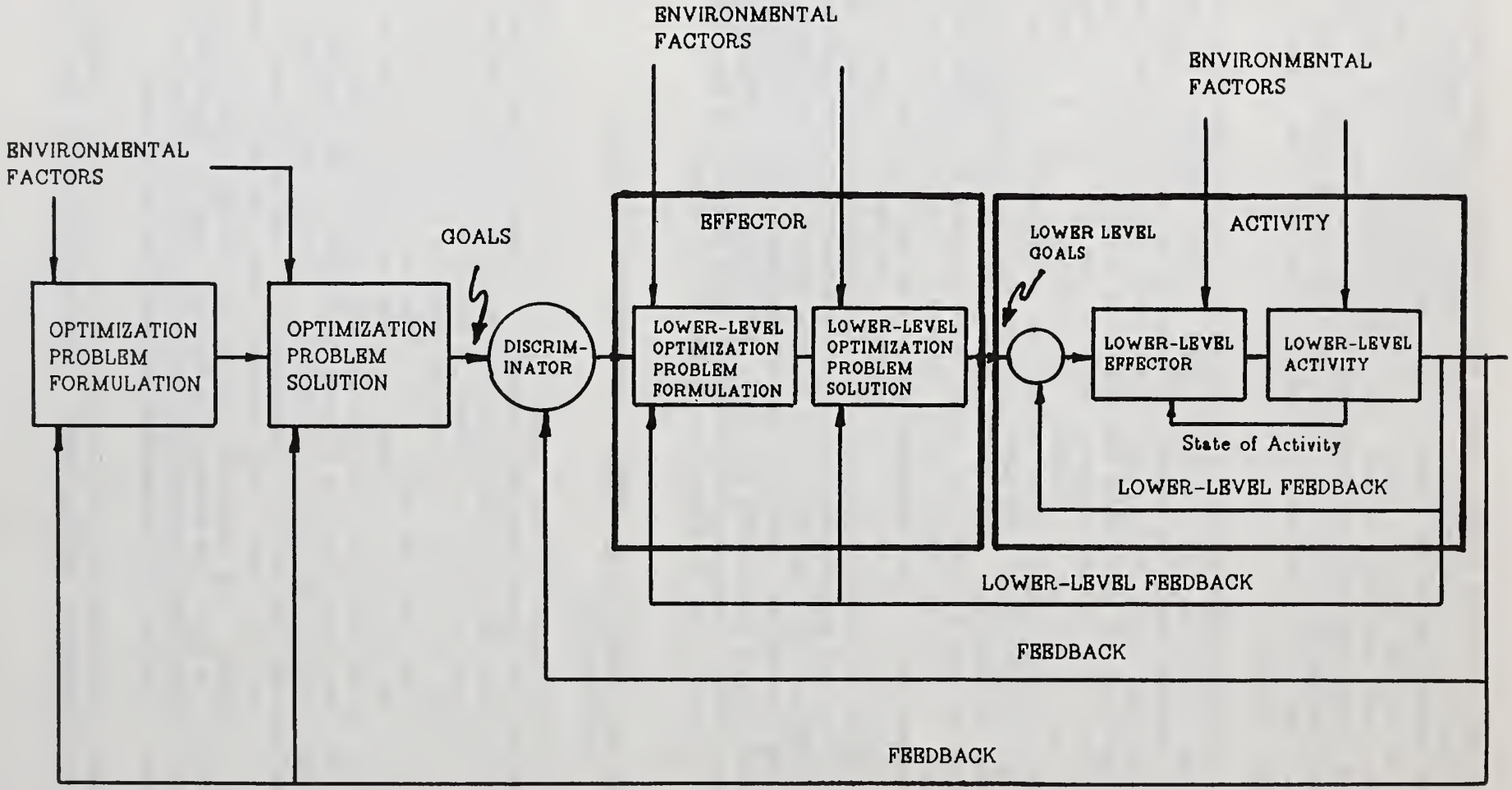


FIGURE 3 – Organization Decision Process Model Expanded to Two Levels

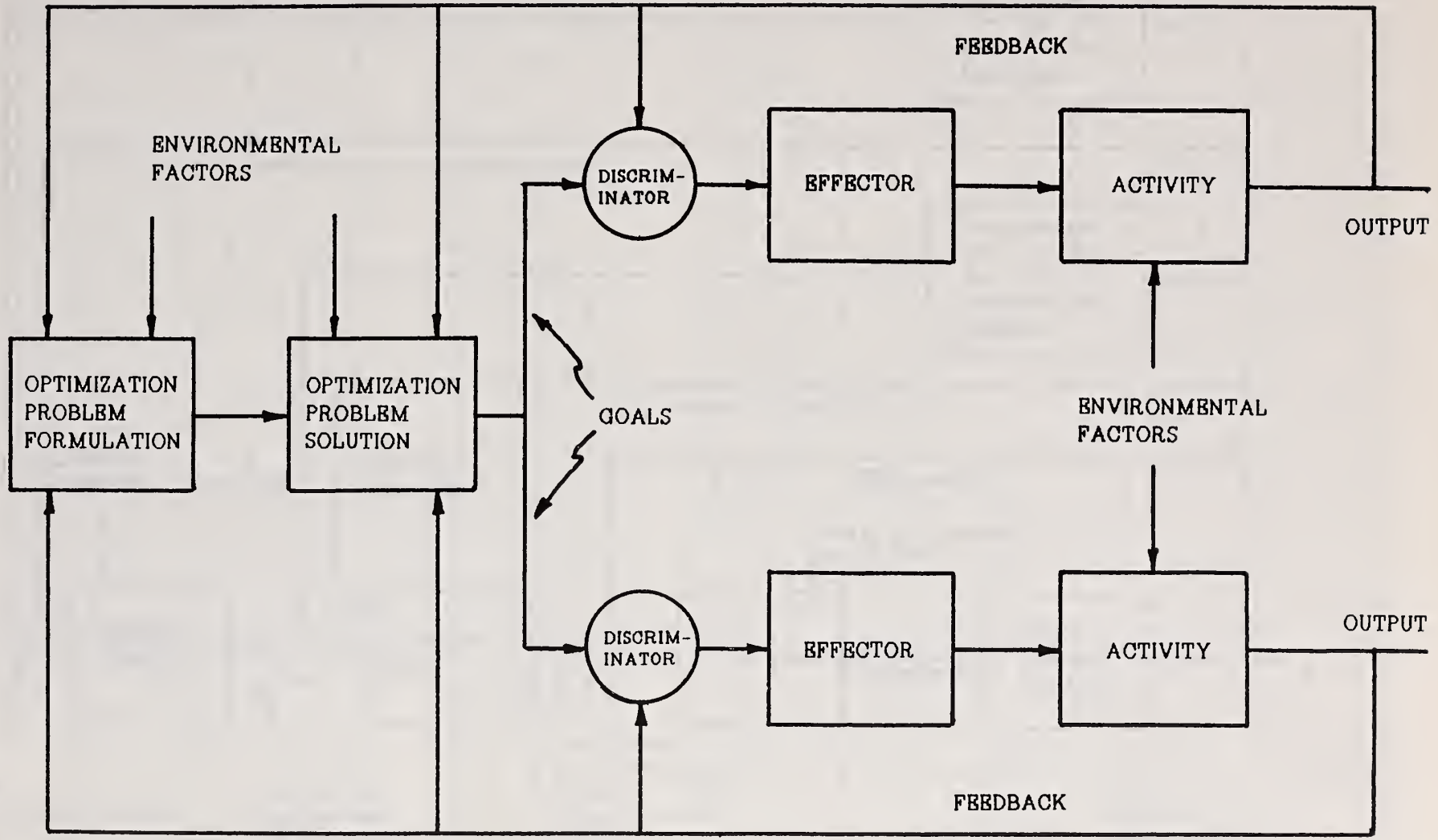


FIGURE 4 – Organization Decision Process Model With a Horizontal Expansion

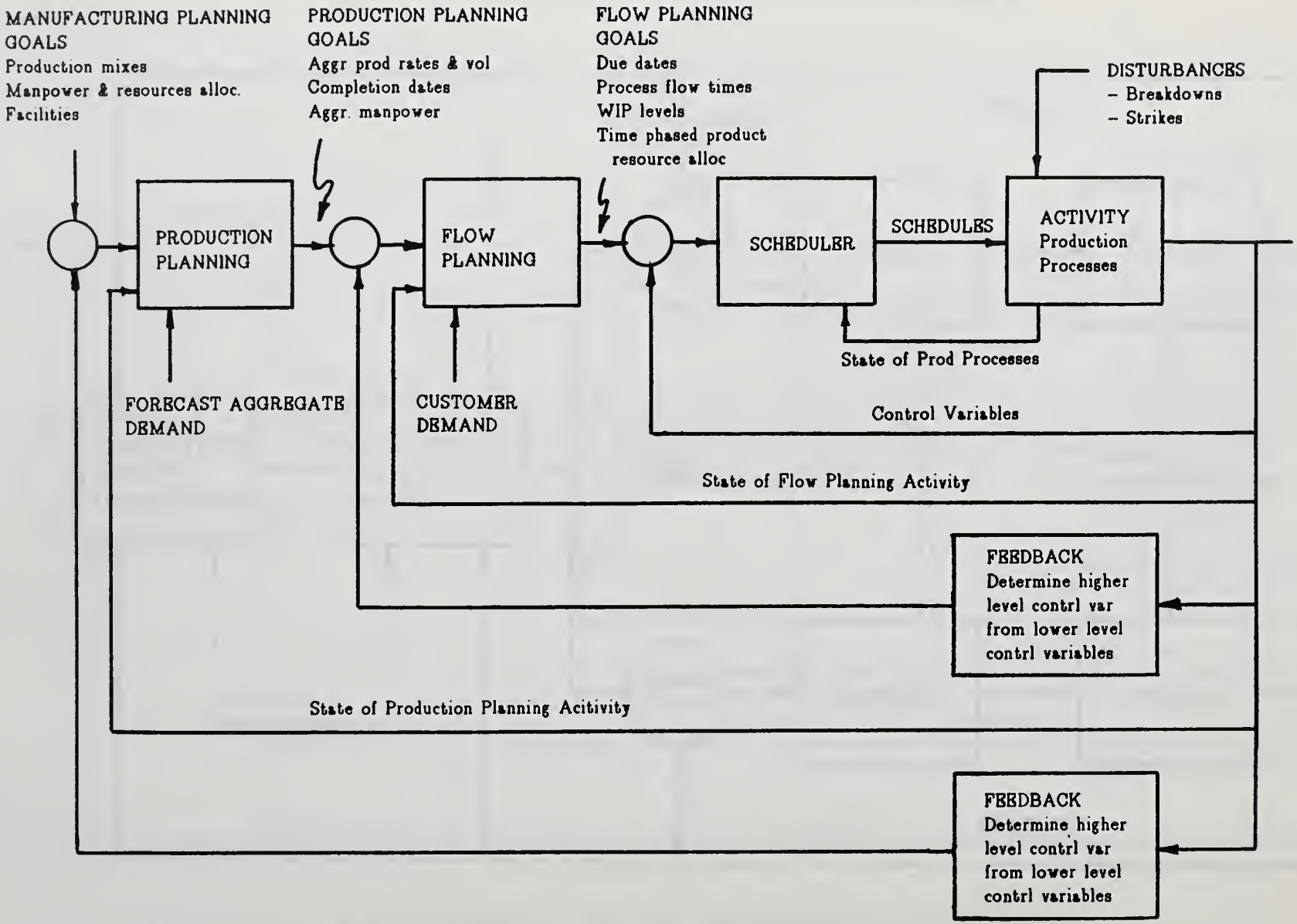


FIGURE 5 – Planning and Management Control Model of the Manufacturing Planning and Scheduling Problem

parameters follow from the manufacturing hierarchy and Abraham and Dietrich (1985). The manufacturing system planners establish mixes of products and the requisite production resources. The production mixes act as goals for the production planning control loop while the facilities' configuration and manpower allocation become constraints in the production planning problem formulation. The production planner formulates the problem using differences from manufacturing goals, forecast aggregate demands, the state of its activity (flow planning control loop), and resource constraints effected by both the manufacturing system planners and the environment. The solution of the problem considers cost minimization and produces goals for aggregate production rates and volumes, completion dates, aggregate inventory costs, and aggregate manpower requirements. The goals (some act as constraints) are applied to the flow planning control loop for achievement. These goals, along with the state of the flow planning activity (scheduling control loop), customer demands, and all constraints imposed by superior levels and the environment, are used by the flow planners to formulate the flow planning problem. Solution of the flow planner's problem trades-off between cost and service level and provides product batch sizes, due dates, process flow times, maximum allowable WIP levels by product, and time-phased product resource availability (raw materials and manpower). These parameters form goals for the scheduling control loop. The scheduler forms and solves a problem which sequences and schedules each batch through the production processes. The scheduling problem is formed from all of the constraints determined by superior level planners, feedback from the product lines, and the goals. The solutions direct the production process activity. Feedback in the scheduling loop provides information on the state of the batches' progress through the production processes and the state of the dynamics of the production processes.

The embedding of the manufacturing levels into the control loops emphasizes goal achievement. Questions concerning how manufacturing levels obtain goals and the consequences of changing plans (goals) can now be addressed. The answers to these questions describe the organizational dynamics resulting from the changes in the decision problem space. Before describing the organizational dynamics, two topics with a direct impact on the dynamics must be discussed. They are the nature of the dynamic problem space and the relationship between planning and operations functions.

The distinction between planning and operation (achievement) functions becomes blurred when describing the achievement of plans. As we can see from the description of the manufacturing levels, planning problems were formulated to yield goals for inferior levels, but the goals were used as commands to obtain superior level goals. Anthony (1965) observes that separating planning functions from operation functions is not straightforward. Indeed, commands resulting from operational decisions at one level of the manufacturing hierarchy are perceived as goals (making it a planning decision) at another level. This requires a trade-off in the decision process characteristics of each effector in the model. A decision process as a planning function should yield stable goals, while as an operation function must revise commands to the activity at whatever rate necessary to compensate for disturbances. The trade-off between these two opposing needs is determined within the context of the organizational dynamics required to cope with the dynamic decision problem space.

Changes in the decision problem space are the forces which trigger the organizational dynamics. The decision problem space consists of the factors which describe the production scenario. The variability in the problem space derives from both influencable and uncontrollable sources. Changes can occur through the environment (uncontrollable), the dynamics of mechanistic elements (some aspects can be influenced), and the choice of decision processes (controllable). The dynamic problem space manifests itself within the model not only as disturbances in the value of the control variables from their goals, but as changes in the structure or dynamics of the manufacturing planning and scheduling decision processes (*i.e.*, planning and operation functions).

We are now in a position to describe the organizational dynamics within the manufacturing hierarchy. The organizational dynamics attributed to goal achievement in a dynamic problem space can touch off two chain reactions. A reaction which propagates up the hierarchy is started whenever a level cannot obtain its goals. Another reaction propagates down the hierarchy whenever plans are changed. Autonomy can be used to effect the bottom up reaction and decentralization to effect the top down reaction. As these reactions are not unique to the manufacturing hierarchy, but occur in any decision hierarchy, the reactions will be described with terminology from the planning and management control framework.

A bottom up reaction is initiated whenever a lower level, unable to meet its goals, defers the decision to a higher level for resolution. The inability of a level to meet goals results either from from changes in the problem space which make the goals infeasible or from insufficient authority to make necessary decisions (autonomy) to react to the changes. The autonomy given a decision maker is the source of the variable effector dynamics mentioned in the previous section. It can be thought of as the scope of decision making rules made available to the effectors. To show how autonomy can effect the decision deferral reaction, consider the actions of an effector to problem space changes that render goal achievement impossible using a particular decision rule. First, the effector searches, within the scope of authority, for a new decision rule which can reinstate goal achievement. Failure to find an appropriate decision rule necessitates deferral of the problem to the next higher level (in particular the adjacent superior level effector) for resolution if goals are to be obtained. This superior level effector, within it's scope of authority, either decides on a new decision rule for the lower level (*i.e.*, rematches the lower-level decision process to the state of the decision problem space) or generates new goals which can be obtained by the lower level with it's current autonomy. Should the superior level effector find it impossible to maintain its own feasibility during this process (*i.e.*, be unable to either find a decision rule or set of goals for the lower level which maintains the superior level's ability to obtain its goals) the decision must be deferred again to the next superior level. The deferral procedure will be repeated up the hierarchy to a level where a resolution which maintains feasibility is possible. If no resolution is possible, the highest level goals for the organization are infeasible and must be changed. Adding autonomy at a level reduces the probability of initiating decision deferrals by maintaining control at that level over a larger range of changes in the decision problem space.

The top down reactions are the results of superior level effectors

adjusting commands to their activities to react to problem space changes. As these commands also form goals, a change of plans (goals) is incurred at the next lower level control loop. Should the inferior level effector find it necessary to change commands to its activity in response to this change in its problem space, more inferior levels will have their problem spaces altered. The plan/command revision propagates down the hierarchy to a level where it becomes unnecessary to change commands in response to the change in plans. Here we see that the values in developing planning decision processes which are robust to changes in the problem space are a reduced number and shorter length of top down chain reactions. It should be noted that a top down reaction can trigger a decision deferral reaction. A bottom up reaction commences from any level that finds no command revision which is capable of achieving the new goals.

Decentralization, since it determines the basic problem structure at each level, can reduce the vulnerability of the hierarchy to top down chain reactions. We want to decentralize the decision processes (problems at each level) in a manner which reduces sensitivity of goal generation (re-solving the problem) at higher levels to uncontrollable changes in the problem space. This will ease the top down reaction effort during plan achievement by eliminating and shortening some of the replanning chain reactions. The structure at each level is determined in part by the parameter aggregation. The choice of parameter aggregation can reduce sensitivity to problem space dynamics. For example, Hax and Golovin (1978) note that aggregate forecasts tend to be more accurate than detailed forecasts. This supports the notion that hierarchical decomposition should consider the effort required to achieve plans as well as the effort required to generate them.

Bounded rationality and dynamic problem space characteristics combine with speed of response requirements to determine the acceptable limits for the frequency of decision deferrals. Response time of the hierarchy is partly a matter of choice, although the upper bound is determined by the most dynamic uncontrollable source of change in the problem space. In addition, decisions usually become more information intensive as sources of change become more dynamic. The response time of the hierarchy increases with the number of levels that must respond. This is in part because of longer information transmission times to higher levels and the increase in the number of decision makers involved in the response. Therefore, in general, faster response times require lower deferral rates and shorter deferral chains. Bound rationality determines the information processing limits of the decision makers. As succeeding lower levels of the hierarchy become more detailed, a decision deferral forces this detail upward, negating the bounded rationality advantages gained in the decomposition. The impact on bounded rationality by of decision deferrals is compounded by the fact that superior level decision makers must respond to the sum of the deferrals from adjacent lower levels. Again, faster response times require lower deferral rates and shorter deferral chains as information processing limits become binding.

Once the upper frequency of occurrence bounds for the two chain reactions in the hierarchy have been determined, decentralization and autonomy are employed to enforce these limits. In general, faster organizational response and increased dynamics of the problem space require greater autonomy and decentralization of detail at lower levels of the hierarchy. The analysis procedure is to determine organizational dynamics

bounds from information handling capacities, response times, and problem space dynamics, and then design the autonomy and decentralization of the organization to meet these bounds. An exception to this procedure must be made when information is unavailable at higher levels of the hierarchy. In such circumstances decentralization of decisions and requisite autonomy must be pushed down at least to the level where the information required is available.

Scheduling in a Dynamic Problem Space

In the previous section organizational dynamics were discussed in general terms. In this section those findings will be applied in detail to the scheduling level. The scheduling level is an operation function which develops schedules as commands to the production processes in pursuit of the goals established by the flow planner. From this description the reason production schedules often change becomes clear. These change with the problem space to continue the production processes on a path to meeting the flow planner's goals. Robust schedules are an attempt to reduce the number of revisions required of the scheduler. Scheduling autonomy is an attempt to reduce the number of deferrals in decision making to the flow planner. First, the scheduling level will be described in detail and then objectives for real-time scheduling developed.

The scheduling level control loop consists of the scheduler, the production processes, goals and constraints from the flow planner (some constraints may be from higher levels), feedback of control variables, and feedback on the state of the production processes (see Figure 6). The structure will be discussed in terms of its effect on the problem space of the scheduler.

Goals received at the scheduling level result from commands generated by the flow planner in pursuit of production planning goals. Some goals simply may be relayed by the flow planner from higher levels. The goals, retaining the manufacturing hierarchy decomposition described earlier, reflect individual product batches and processes. Table I lists the possible goals. These goals will be included in a schedule-generating decision problem. Some goals represent objectives and others inviolable constraints. To guide schedule generation, the scheduler also may be given a performance philosophy to follow in pursuing these goals, such as minimizing tardiness or total flow times by product, or balancing utilization. In many cases the performance philosophy takes a back seat to just finding schedules that realize the flow planners goals (*i.e.*, finding feasible schedules).

Constraints in the scheduling decision problem space have a multiplicity of sources. Some are formed from the goals as noted, while others result from the state of the production processes. Facility configuration constraints are relayed down the hierarchy from the manufacturing system planning level. The flow planner places additional constraints on the activity as a result of its own problem formulation. A list of possible constraints appears in Table I.

From these goals and constraints the scheduler forms an appropriate problem to develop schedules for the production processes. The scheduler's

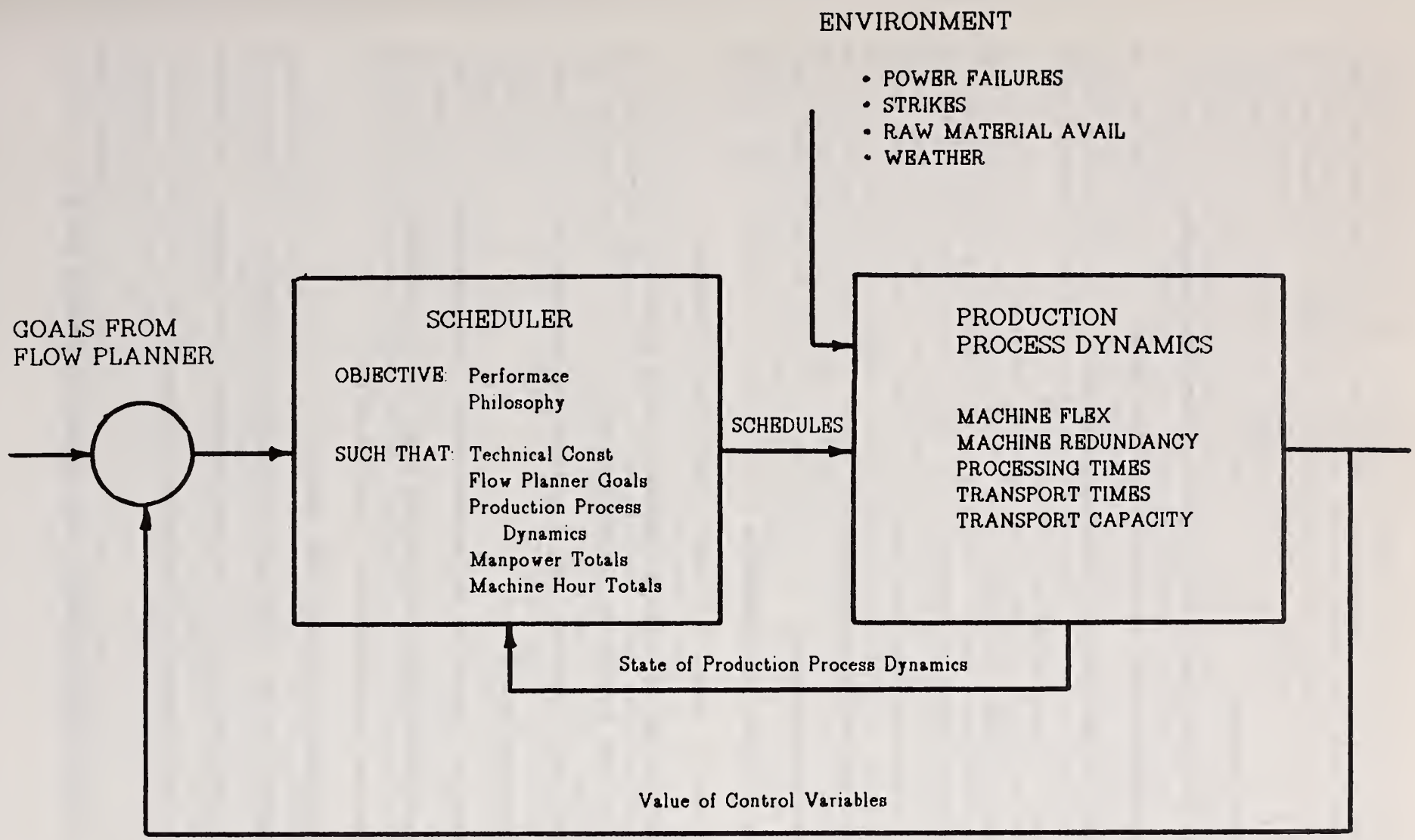


FIGURE 6 - Planning and Management Control Model of the Scheduling Level

GOALS (From Flow Planner)

- Batches of determined sizes finished within time horizon
- Maximum Allowable WIP levels by product or batch
- Flow times of batches on individual processes
- Batches of determined sizes with due dates
- Manpower available for batch
- Manpower available for process
- Machine time available for batch
- Machine time available for process

CONSTRAINTS (Framework Source)

+ Product Process Characteristics
    - Transport capacity by batch (production processes)
    - Machine set-up times for products (flow planner)
    - Technical constraints (flow planner)
    - Transport times of batches (production processes)
+ Facilities and Resources
    - Machine flexibility at process steps (prod.processes)
    - Machine redundancy at process steps (prod. processes)
    - Machine hours available (flow planner)
    - Man-hours available (flow planner)
    - Machine set-up by process (production processes)
    - Material availability at each process (flow planner)
    - Transport capacity (production processes)

TABLE I - GOALS AND CONSTRAINTS FOR SCHEDULING PROBLEM SPACE

problem formulation is subject to change via top down replanning chain reactions and can be a source of decision-deferral chain reactions. Here we can see that the amount of autonomy at the scheduling level manifests itself as the scheduler's authority to reformulate the schedule-generating problem. The dynamics of the scheduling decision problem space determine the frequency and degree of reformulation required to produce feasible schedules. Table II lists the types of dynamics in the problem space of the scheduler and their sources from the scheduling level structure. The production processes change as a result of internal stochastic disturbances or environmental factors (*e.g.*, power failures, strikes, etc.). Goals may be revised by the flow planner in response to environmental factors or higher level fiat (*e.g.*, reduction in budgets). As these dynamics change the schedule generating problem parameters, the scheduler may have to change other parameters under his control to maintain feasible schedules.

A large part the schedule-generating problem structure was determined by the choice of the manufacturing decomposition. Note that in Tables I and II, cost and demand parameters are absent. These parameters are accounted for at higher levels and their effects are contained in the scheduler's goals. In addition the scheduler is given predetermined batch sizes, flow times, and due dates. Time horizon for the problem usually runs from a day to a couple of weeks. The schedule-generating problem's decision variables are the times to schedule the batches on the machines. Solution of the problem yields a set of times for each job on each machine which meet the constraints of facility configuration, WIP levels, batch sizes, due dates, and machine and manpower dictates. Should multiple solutions be found, the scheduler chooses among alternatives using the performance philosophy.

From Table II and Figure 6 it becomes obvious that changes in the scheduling problem space manifest themselves as changes in the constraints of the schedule-generating decision problem. Some constraint changes render the current schedule infeasible, requiring re-solving the problem with the new constraints. Other constraint changes can render the whole schedule generating-problem infeasible (*i.e.*, the current problem formulation adjusted to reflect changes in the problem space contains no feasible solutions). In this case the scheduler, within his autonomy, must reformulate the problem by adjusting the remaining constraints to readmit feasible schedules. If the scheduler's autonomy is insufficient to reformulate a feasible problem, either infeasible schedules are employed or the problem is deferred to the flow planner. To keep the frequency of deferrals within acceptable bounds the authority to change constraints needs to be matched to the scheduling-problem space dynamics.

The amount of autonomy given the scheduler must be evaluated for each manufacturing and planning hierarchy. Generally, it is preferable to retain the scheduling decision at the scheduling level. This of course is subject to the amount of autonomy necessary to allow the scheduler to find feasible schedules. At some point additional autonomy can become more detrimental to the organizational pursuit of goals than the deferral of the decisions. The autonomy afforded the scheduler to change the schedule generating decision problem also provides the scheduler with requisite authority to make the actual changes in the production scenario. The end result of these actions are changes in the goal states obtained by the production processes. A general rule of thumb might be to grant the scheduler autonomy up to the

PROBLEM SPACE DYNAMICS (Source)

Machine failure (production processes/environment)

Machine repair times (production processes/environment)

Transport failure times (production processes/environment)

Process times (production processes)

Due dates (flow planner/environment)

Batches added (flow planner/environment)

Manpower availability (higher level via flow planner/envir.)

Raw material availability (production processes/environment)

Machine availability (higher level via flow planner)

WIP levels (flow planner/higher levels)

Machine set-up times (production processes)

Transport times (production processes)

Transport capacities (production processes)

TABLE II - Types of Problem Space Dynamics for Scheduler

point where the flow planner's decision problem (*i.e.*, the problem that determines commands which achieve the production planning goals) requires resolving. This slack in the flow planner's commands would have to be supplied by the flow planner. This rule gives the scheduler autonomy to react to disturbances while not interfering with the flow planner in achieving his goals. The autonomy bound requires the flow planner to make decisions when his goals are in jeopardy.

The scheduler acts as the controller of the schedule control loop. Up to this point, schedule generation characteristics have been based on decentralization and autonomy required to bound the frequency of replanning and decision-deferral chain reactions. As the source of control commands to the production processes, the scheduler must also generate schedules which are matched to the production process dynamics. The dynamics are represented by the set-up times, process times, machine availability, etc. These factors, along with information transfer dynamics, place an upper limit on the rate of rescheduling. Robustness in the schedules is desirable at least to the point that it impacts the rescheduling limit. Robust schedules reduce the need of the scheduler to adjust commands, which also reduces the possibility of initiating the two chain reactions. Another consideration which effects the speed with which schedules must be generated is the problem space dynamics. From the standpoint of the production process dynamics, there is no need to generate schedules faster than the rescheduling limit. However, the problem space dynamics can cause a need to reschedule at a rate faster than this limit. For example, equipment breakdowns can cause schedules to become outdated before these are issued.

SUMMARY

The production planning and scheduling problem can be approached as a planning and management control problem. The planning and management control framework not only considers the planning of goals, but their achievement within a dynamic environment. Use of the framework emphasizes the organizational dynamics of the manufacturing hierarchy. This allows the analysis of the production planning and scheduling problem within a dynamic problem space on a real-time basis.

The dynamic problem space causes two chain reactions within the manufacturing hierarchy. Decentralization is used to bound the replanning reaction and autonomy is used to bound the decision-deferral reaction. Decentralization determines the decision process structure at each level. The decision structure needs to consider the effort required in achieving plans as well as the effort in generating plans. This can be accomplished by decreasing the sensitivity of higher levels to changes in the problem space, reducing the need to replan. Autonomy allows the decision maker at each level latitude in changing the decision process as a means to cope with the dynamic problem space. Autonomy is used to decrease the number of decision making deferrals.

The dynamic problem space causes changes in the constraints of the schedule-generating problem, which results in rescheduling. To lessen the number of rescheduling deferrals to the flow planner, autonomy in the form of authority to change the scheduling problem constraints should be granted the scheduler. A possible rule for the scope of this authority is to allow

the scheduler to make changes within the slack present in the the flow planner's commands (to the scheduler) determined to achieve flow planning goals. Rescheduling needs to be matched to the production process dynamics. This is accomplished through (1) generating robust schedules which allow for larger problem space changes before rescheduling and (2) matching the time to generate schedules to the problem space dynamics. Real-time scheduling decision aids must respond to these needs.

REFERENCES

Abraham, C. and Dietrich, B.; "Discrete Manufacturing Taxonomy"; working document, IBM Research, Yorktown Heights, N.Y., 1985.

Abraham, C., Dietrich, B., Graves, S. and Maxwell, W.; "A Research Agenda for Models to Plan and Schedule Manufacturing", July, 1985.

Ackoff, L. Russel; "Towards a System of Systems Concepts"; Mgt. Sci., Vol. 17, No. 11; July, 1971.

Andersson, H., Axsater, S. and Jonsson, H.; "Hierarchical Material Requirements Planning"; Int. J. Prod. Res., Vol. 19, No. 1, pp 45-57, 1981.

Anthony, R. N., Planning and Control Systems - A Framework for Analysis; Copyright 1965 by the President and Fellows of Harvard, Boston.

Axsater, S.; "Aggregation of Product Data for Hierarchical Production Planning"; Operations Research, pp. 744-756, 1981.

Bitran, G.R., Haas, E.A., and Hax, A.C.; "Hierarchical Production Planning: A Single Stage System":Operation Research, vol. 29, pp. 717-743; 1981.

Bitran, G.R., Haas, E.A., and Hax, A.C.; "Hierarchical Production Planning: A Two Stage System"; Operations Research, Vol. 30, No. 2, March-April, 1982.

Bitran , G.R. and Hax, A.C.; "On the Design of Hierarachical Production Planning Systems"; Decision Sci. vol. 8, pp. 46-57, 1977.

Flamholtz,E.G., Das, T.K. and Tsui, Anne S.; "Toward an Integrative Framework of Organizational Control"; Accounting Organizations and Society, Vol. 10, No. 1, pp 35-50; 1985.

Fox, M.S., Allen, B.P., Smith, S.F. and Strohm, G.A.; "ISIS: A Constraint-Directed Reasoning Approach to Job Shop Scheduling System Summary"; Technical Peport CMU-RI-TR-83-8, The Robotics Institute, Carnegie-Mellon University; 1983.

Gershwin, S.B., Hildebrant, R.R., Suri,R., and Mitter, S.K., "A control Theorist's Perspective on Recent Trends in Manufacturing Systems", Presented at the 23rd IEEE Conference on Decision and Control, Las Vegas, Nevada, December, 1984.

Hax, A.C. and Golovin, J.J.; "Hierarchical Production Planning" in Studies in Operations Management, A.C. Hax (ed.), North Holland, Amsterdam; 1978.

Hax, A.C. and Meal, "Hierarchical Integration of Production Planning and Scheduling" in Studies in Management Sciences Vol. 1 Logistics, ed. M.A. Geisler, pp 53-69; North-Holland-American Elsevier; 1975.

Maxwell, W., Muckstadt, J.A., Thomas, J., and VanderEecken, J.; "A Modeling Framework for Planning and Control of Production in Discrete Parts Manufacturing and Assembley Systems", Interfaces; vol.13, pp. 92-104, 1983.

Meal,H.C.,; "Putting Production Decisions Where They Belong"; Harvard Business Review, March-April, pp. 102-110, 1984.

Merchant, K.A.; Control in Business Organizations; Pitman Publishing Inc., 1985.

Midduagh, J.K. and McPherson,R.F.; "A Conceptual Framework For Planning and Control Using A Generic Decision Process Approach"; Working Paper, Colgate Darden Graduate School of Business Administration, University of Virginia, 1986.

Young,S.; Management: A Systems Analysis;Scott, Foresman and Company, Glenview, Illinois; 1966.

HIERARCHIES OF SUB-PERIODS IN CONSTRAINT-DIRECTED SCHEDULING

Mitchell S. Steffen
Industrial Automation Division
The Charles Stark Draper Laboratory, Inc.
555 Technology Square
Cambridge, Massachusetts 02139

Timothy J. Greene, Ph.D
Dept. of Industrial Engineering and Operations Research
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061

INTRODUCTION

Today's dynamic markets require greater flexibility on the part of manufacturers. However, many operations in process industries were designed for highly efficient production of a few commodity products, and therefore often lack the flexibility needed to follow increasingly variable market demands. In these situations, the operation's scheduling system must help identify and exploit any opportunities for adapting to changing circumstances.

Computerized scheduling systems for process operations typically make decisions by solving a mathematical model of the operation's costs and constraints [1], [2]. Such a scheduling system seeks to optimize a cost function that represents the most efficient way to run the operation. However, because the mathematical model uses a static representation of the operation, this approach does not always help the operation's management identify and use the available flexibility to react to the opportunities and problems that frequently arise.

The research reported here investigated an alternative approach to developing scheduling systems for batch process operations consisting of parallel processors that are subject to preference, buffer inventory, and sequencing constraints. This approach derives from research in Artificial Intelligence (AI) and applies hierarchical planning and constraint-directed heuristic search to develop a prototype scheduling system for a specific case study problem. Using a constraint satisfaction model of a process operation allows an automated scheduling system to specifically search for available degrees of freedom to use in devel-

oping schedules and in providing decision support to an operation's management.

To reduce problem complexity, this research investigated the decomposition of the scheduling problem along the time-line into sub-periods. The reduction of complexity achieved by this approach permitted schedules for each processor to be developed in parallel, while enabling the prototype system to fulfill constraints whose application was time-dependent. The use of a planning hierarchy helped anticipate and reduce interaction between sub-periods.

The prototype's performance was evaluated by testing it with actual production data and comparing the results with schedules created by the human scheduler. Adherence to constraints formed the basis of comparison. The schedules created by the prototype compared favorably with the human scheduler's, suggesting that this approach could potentially provide the means for developing an operational scheduling system.

PROBLEM DESCRIPTION

The following problem description provides the background that gave rise to the research presented in this paper. While the prototype scheduling system was developed for a specific case study, the prototype's structure can apply to a wider range of scheduling problems, particularly since parallel processor problems occur frequently in process industries [1].

The case study production operation is a department within a large bulk material process operation. The facility operates on a continuous, 24-hour per day, seven days per week basis. The department converts a liquid raw material into powder form for down-line departments, hereafter referred to as "consumers". The addition of special ingredients gives rise to a variety of final products. To meet consumer demand, 10 to 15 different products, out of about 150 possible products, are in prodution at a given time.

While the department has three stages of parallel processors, because of technological and organizational constraints they are treated from a scheduling viewpoint as a single set of parallel processors. Output buffers store completed batches. Each buffer supplies one down-line consumer that draws the product from the buffers at a continuous rate. The consumers make no attempt to match this rate with the processor's batch size. Figure 1 gives a schematic of the "logical" system. The case study is described in more detail in [3].

In general, any processor can produce any of the products required by any of the consumers, but slight variations between processors result in the consumers preferring specific processors as their product source. Producing batches on non-preferred processors causes down-line quality problems, so consistently satisfying these preferences is the human

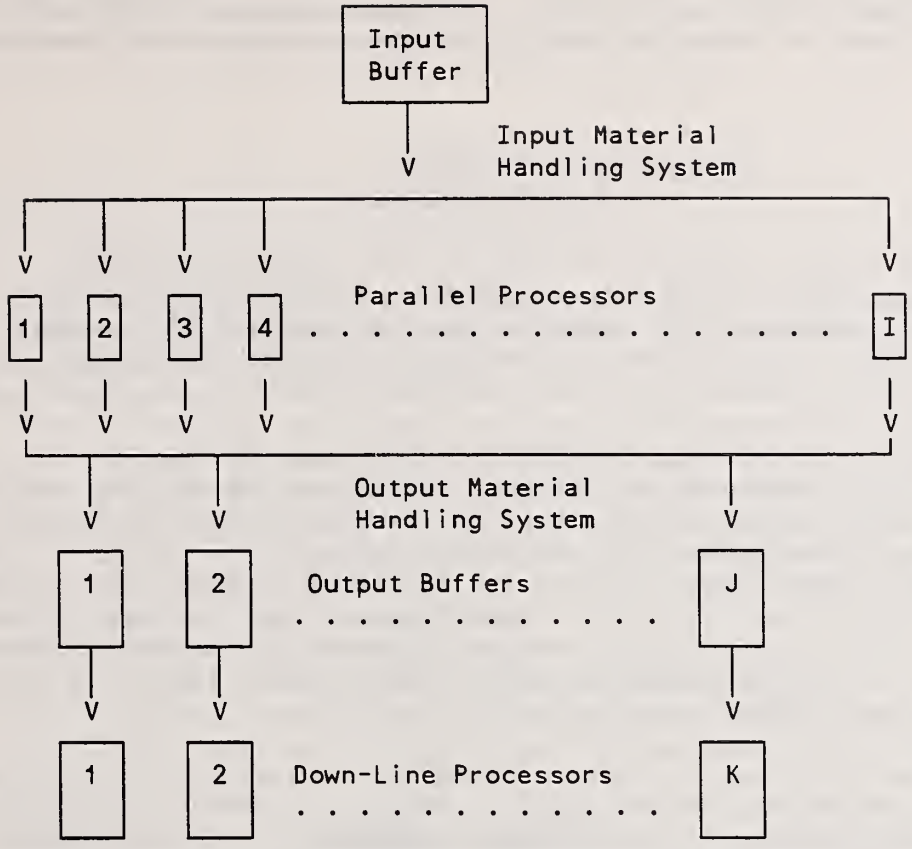


Figure 1. Schematic of System From Scheduling Viewpoint

scheduler's primary objective. The set-up costs resulting from allowing an output buffer to be exhausted are prohibitive. Therefore, another important objective of the human scheduler is to maintain a sufficient safety stock in each buffer. The scheduler also considers several secondary objectives, such as minimizing material handling and minimizing the labor required to execute the schedule.

Several constraints limit achievement of these objectives. Incompatibilities between products prohibit certain sequences of products on the same processor. The output buffers have fixed, finite capacities. Power consumption and organizational needs require that processor batch cycle times begin and end at prescribed times as given in a standard schedule form, without preemption. Certain processors are not equipped to make products requiring extra ingredients. The output material handling system becomes congested when too many batches are scheduled in processors on the opposite side of the facility from the destination buffer. Exactly 100 percent of processor capacity must be scheduled.

Maintaining buffer safety stocks and observing buffer capacity and product sequencing restrictions are all examples of time-dependent constraints. They are time-dependent in that the specific parameters asso-

ciated with a constraint at time t are dependent on the scheduling decisions made prior to that time. The exact application of such a constraint cannot be determined until the scheduling decisions preceding it have been made.

RELATED RESEARCH

A variety of views on how to apply AI methods to scheduling problems have been suggested (for examples see references [4] - [14]), and several major corporations have announced undertaking research efforts in this area. However, no reports of operational systems have appeared to date.

The first research on AI-based industrial scheduling systems to appear in the literature concerned the job-shop scheduling system ISIS (Intelligent Scheduling and Information System) [15] - [21]. ISIS was developed by researchers at Carnegie-Mellon University, and tested on simulated data derived from a Westinghouse turbine blade production facility. Currently, ISIS is undergoing conversion by Westinghouse for potential implementation at pilot facilities [22], [23] . A successor system called PHOENIX has been initiated [24].

The approach used in ISIS is to model the variety of objectives in the job-shop as constraints on the schedule. In addition, ISIS considers a large number of technological constraints. Scheduling is performed by searching for decisions that satisfy the variety of constraints considered by the system. When a problem is overly constrained, ISIS finds a feasible solution by heuristically and selectively relaxing constraints. Because of the great complexity of the problem, ISIS uses hierarchical planning to break the problem into manageable pieces. Each layer in the hierarchy solves a specific abstraction of the problem and creates constraints to limit search at lower layers.

Several other AI-based scheduling systems have been developed. Although they do not specifically address process operations, they can provide useful insights into AI methods and their potential applications. Engineers at Lockheed-Georgia developed a rule-based system for controlling hoist movements in an automated paint and process line [25]. Fox and Kempf [26] used an opportunistic approach to schedule robotic assembly operations. Fukumori [27] used timing constraints to schedule pass-through relationships and arrival and departure times for passenger trains. Glover et al [28] used a rule-based approach to develop employee schedules for fast-food restaurants. NUDGE [29] uses script-based planning and debugging to maintain an appointment calendar while monitoring the progress of worker's goals and alerting managers of upcoming deadlines. TABS [30] uses heuristic backtracking to schedule office meetings. ISA [31] uses a rule-based approach to provide firm delivery dates for customer orders by reserving inventory and production capacity on an aggregate plant level. PEPS [32] uses a rule-based approach to

select dispatching rules for a single machine with a queue of parts. DEVISER [33] generates parallel plans for spacecraft which must schedule several instruments during planetary fly-bys.

APPROACH TO THE PROBLEM

Approaches used in AI research in planning include [34]: hierarchical, non-hierarchical, script-based or skeletal, and opportunistic. A hierarchical planning and constraint-directed heuristic search approach was chosen for several reasons:

- A variety of objectives and constraints apply when developing schedules, and trade-offs must be made when constraints conflict.
- The human scheduler develops schedules in a hierarchical fashion, working from abstractions of the problem to increasingly detailed plans. While the prototype does not necessarily have to mimic the human scheduler, it should take advantage of whatever insights the scheduler has.
- The results from the ISIS research suggest that this approach naturally models many scheduling problems.
- Stefik et. al. [35] recommend using this approach for problems with large search spaces and interacting subproblems.

In using this approach, all objectives are recast as constraints. The total set of constraints then defines the set of feasible schedules, which is rated by heuristics. For each level of the hierarchy, a subset of the problem constraints actually apply, and create a search space. If the problem becomes overly constrained, constraints are heuristically relaxed.

The objective of this research was to build and test a constraint-directed heuristic search based prototype scheduling system to determine if the approach is feasible for production operations like the case study. Before beginning work on an operational scheduling system, a prototype scheduling system should be built and tested. The prototype serves two purposes [36]:

1. It provides a means for the system developers to check their understanding of the problem.
2. It provides a test bed to determine if the solution approach is feasible for the problem before excessive resources have been committed to system development.

The second purpose is particularly important in cases where the approach taken is experimental in nature.

The prototype scheduling system only addresses a subset of the objectives, constraints, methods, and other parameters found in the case study's system. Limiting the scope of the prototype in this manner allowed attention to be focused on the fundamental design issues associ-

ated with representing and solving the scheduling problem using AI methods. This subset must be explicitly defined so that realistic performance criteria for the prototype system can be set and objectively evaluated. A complete description of the prototype's scope is given in [37].

The prototype scheduling system was written in Virginia Tech Prolog [38], which runs in an interpreted mode on a VAX 11/780 under VMS. Several features of Prolog allowed rapid development of the prototype, including:

- Facility for expressing arbitrary relationships between objects as predicate clauses.
- Pattern matching capabilities.
- List processing functions.
- Automatic backtracking of program control when a line of execution fails.

The prototype makes extensive use of these features to develop and search the sets of feasible schedules. Clocksin and Mellish's text [39] provides a good introduction to Prolog. To supplement the facilities of Prolog, some relational table manipulation and execution tracing routines from the GUESS/1 [40] expert system development shell were used.

SOLUTION DESIGN

Before building the prototype, a general framework for scheduling the process was developed by analyzing the case study problem. The purpose of this framework was to anticipate the ultimate needs of an operational scheduling system. In other words, through an engineering analysis of the problem, the prototype design was based on what "should be" as well as what "currently is".

The development of an automated scheduling system represents an opportunity for the operation's management to step back and analyze the current assumptions and procedures, and from this analysis initiate improvements. Developers of the system should facilitate this process and incorporate it into their design. Beyond these issues, the existing operating procedures, scheduling methods and data sources of this case study could not provide a sufficient basis for an automated scheduling system for the following reasons:

1. The human scheduler's decisions are frequently based on negative feedback from management or downline consumers; an automated system must take a more global view of the problem because of the impact that scheduling decisions have on down-line operations.

2. Certain operating policies that the human scheduler has no control over could be changed as a result of a problem analysis.

3. Some important information concerning the operation is difficult to obtain or analyze manually, and therefore has little impact on current decision making, but could easily be incorporated into a computer-based system.

From this analysis, a decision hierarchy emerged:

1. Production Planning: Balance capacity, inventory and demand.

2. Loading: Assign batches (jobs) to processors.

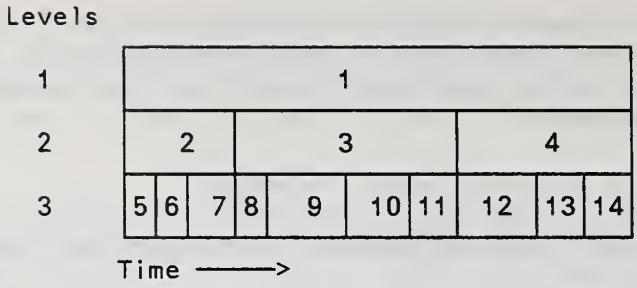
3. Sequencing: Determine order of batches loaded on each processor.

The first level is analogous to an aggregate production planning problem [41], but on a much shorter time scale. The second level focuses on maintaining the consistency and stability of the schedule. The third level assures that time-dependent constraints are satisfied. In constrast to mathematical programming-based hierarchical scheduling models, the higher levels do not dictate bounds to the lower levels, but rather provide "guidance" constraints that focus lower level searches in areas most likely to contain good solutions.

For the second and third levels of this hierarchy, the resulting search space complexity remained great enough to require further decomposition. The ISIS and PHOENIX research suggest two candidate decomposition methods: order-based and resource-based. Order or lot-based decomposition, where jobs are scheduled one at a time in priority order, would not satisfy the time-dependent constraints of the problem addressed in this research without large amounts of backtracking. Because the batches for each consumer are produced by multiple processors, the sub-problems created by a resource-based decomposition would have high levels of interaction. Process scheduling problems like this case study need an approach that considers all processors and batches simultaneously, in parallel. However, such an approach must deal with the complex search space created by these constraints.

The approach taken in this research was to further decompose the problem along the time-line, into sub-periods. The primary concern with this approach is the possibility of sub-problem interaction. However, by using constraints to define sub-period boundaries, they could also be used to communicate interactions across these boundaries [42], [43] .

Figure 2 shows the relationship between the hierarchy's levels and the sub-period definitions for a hypothetical scheduling problem. Defining second level sub-period boundaries by capacity or demand changes creates simpler problems of constant capacity and demand. Defining third level sub-period boundaries with time-dependent constraints insures that all decisions which must precede a constraint application have been made.



(1-14 give a hypothetical sub-problem solution order)

Levels	Sub-period Boundary
1. Balance Capacity, Inventory & Demand	End of planning horizon
2. Loading	Capacity or demand change
3. Sequencing	Buffer safety stock constraints

Figure 2. Decision Hierarchy and Sub-period Definitions

The use of a "hierarchy of sub-periods" framework requires the system designer to make several problem specific decisions:

1. Sub-Problem Execution Order: The choice is between a "breadth-first" order (as shown in Figure 2) or a "depth-first" order. For the case study, the design used a breadth-first order in anticipation of less total backtracking between sub-problems.

2. Length of Planning Horizons: In general, the lower the level in the hierarchy, the shorter the required planning horizon. However, the case study problem required all the planning horizons of each level to be equally long to be certain that the system did not "paint itself into a corner" by not anticipating the effects of the time-dependent constraints.

3. Sub-Period Boundary Definitions: The criteria for defining sub-period boundaries must be designed so that sub-problem interactions are minimized. In this research, problem constraints provided effective criteria. If the criteria differs between levels of the hierarchy, the sub-period boundaries will probably not coincide between levels. However, as shown in Figure 2, the problem structure may dictate that the criteria for one level dominate the criteria for the next lower level. Boundaries may be determined before or during the solution procedure. For the case study, all information for determining second level boundaries is available in the input data, while each third level boundary is determined dynamically, after the preceding sub-problem is solved.

PROTOTYPE IMPLEMENTATION

Prototype development occured incrementally, until its performance demonstrated the feasibility of the approach. Complete implementation of the design framework described in the preceding section would be a major project and beyond the scope of this research. The experience gained from developing the prototype could serve in further refining the design of an operational system. The following section outlines the structure of the actual implementation. A more detailed description of the prototype scheduling system is given in [44].

Using the framework outlined in Figure 2, search strategies were developed for the sub-problems of each level. These strategies were derived from study of the human scheduler's methods and rules-of-thumb. In the interest of obtaining fast feedback, simple heuristics were used to implement these strategies. In most cases, to take advantage of Prolog's features, depth-first search in priority and preference order was used.

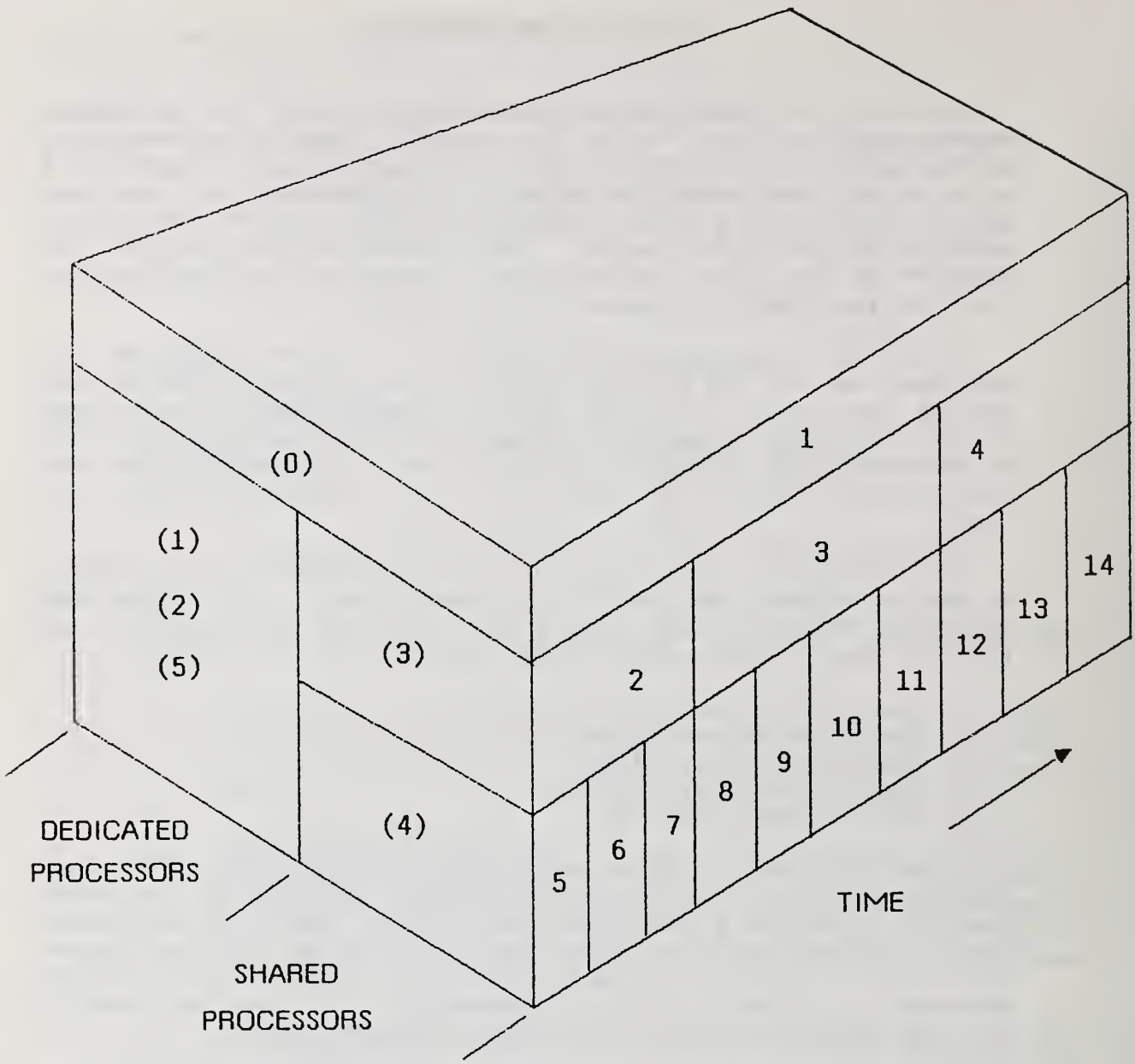
As implemented, the prototype scheduling system develops schedules by working through five hierarchical planning layers. Each layer adds increasing detail to the schedule and constrains lower layers. The planning layers in hierarchical order are:

1. Dedicated processor scheduling
2. Special batch scheduling
3. Shared processor loading
4. Shared processor sequencing
5. Batch demand scheduling.

Dedicated processors are those whose total production capacity for the scheduling period is assigned to one consumer. Because only one batch type is assigned to a "dedicated" processor, no sequencing is needed. Shared processors are those whose production capacity is divided between two or more consumers, giving rise to separate loading and sequencing problems. This "dedicated-shared" decomposition is a heuristic used by the human scheduler to further reduce problem complexity.

Special batches must be assigned to a specific processor at a specific time. Batch demand consists mainly of miscellaneous batches that have no timing or preference constraints. The position of layers two and five in the hierarchy reflects a plan repair or "debugging" approach [45] in that they sometimes must undo decisions made in preceding layers. This heuristic approach greatly simplified the logic of layers one, three and four by allowing them to ignore special cases. At first glance, the special batches would appear to provide a basis for using opportunistic search [46]. However, these batches are peripheral with regard to satisfying the major objectives of the problem, so little would be gained by searching from the fixed points they provide.

Figure 3 depicts the relationship between the solution design and the prototype implementation. A simple rule-based system with some



Implementation Hierarchy

(0) Balance Capacity, Inventory And Demand (Given In Input Data)

(1) Dedicated Processor Loading (And Implicit Sequencing)

(2) Special Batch Loading And Sequencing

(3) Shared Processor Loading

(4) Shared Processor Sequencing

(5) Miscellaneous Batch Loading And Sequencing

Figure 3. Prototype Implementation

arithmetic sub-routines could have provided a workable prototype to simulate the human scheduler's logic for level zero. However, for the available test data, manually simulating the logic in the input data was sufficient for this research.

The constraints (including reformulated objectives) addressed by the prototype include:

1. Preference for specific processors by consumers
2. Output buffer safety stocks
3. Output buffer capacities
4. Sequencing of incompatible batches
5. 100 percent utilization of production capacity
6. Fixed batch cycle start times.

The prototype relaxes the buffer safety stock and preference constraints when search spaces are overly constrained, in that order. The buffer safety stock constraint is relaxed by incrementally reducing the minimum buffer safety stock level (until an absolute minimum level is reached). The preference constraint is relaxed by including non-preferred processors in the search space. All relaxation is done locally, on a case by case basis. When developing a search space, each constraint is applied at its initial or tightest level. When a constraint relaxation occurs, the relaxation applies only to the specific batch that is overly constrained. A benefit of this approach to constraint relaxation is that when a specific sub-problem requires unusually large amounts of relaxation, the system user has an inidication that this time period requires special attention.

Data structures for the knowledge base were developed as needed to support the hierarchical planner. This is the opposite of the approach used in developing expert systems [36]. The bulk of the work in building an expert system is devoted to the knowledge base; a previous application could provide the inference engine. Reference [37] provides details on the knowledge base structure and contents.

RESULTS

Evaluation of the prototype scheduling system was based on a comparison with schedules created by the human scheduler. Schedules created by the human scheduler represent the baseline of performance expected from the prototype. The management of the production operation did not use quantitative measures of schedule quality; this research's scope did not include the determination of objective criteria for schedule evaluation.

Tests of the prototype scheduling system used eight days of actual data from the case study for which the human scheduler provided detailed notes on how he created the schedules. These eight days included many of the events that the scheduler normally must deal with, such as:

changes in consumer demand type or quantity, processor status changes (on or off-line), requests for special batches, and changes in batch demand type or quantity. Adherence to the six constraints addressed by the prototype formed the basis for comparison between the prototype and human scheduler. Each schedule was analyzed to determine if constraints were adhered to, and if not, the degree to which the constraints were relaxed. The following paragraphs summarizes the results of this comparison.

Adherence to Non-Relaxable Constraints: Both the prototype and the human scheduler observed all non-relaxable constraints (constraints 3-6), with one exception. The exception was that the human scheduler exceeded the output buffer capacity constraint in 11.5% of the batches scheduled, while the prototype did not relax this constraint at all. While the output buffers do have a fixed capacity, an implicit overflow capacity exists in the output material handling system. The human scheduler takes advantage of this implicit overflow capacity in order to give himself a larger safety stock. The prototype system has no logic for relaxing this constraint. However, its performance in maintaining the safety stock constraint suggests that it may not need to use this degree of freedom as the human scheduler does in this situation.

Adherence to Relaxable Constraints: Both the prototype and the human scheduler relaxed the preference constraint in only 0.1% of the batches scheduled. Apparently there was little contention for resources during the period from which the test data were taken. The prototype relaxed the output buffer safety stock buffer constraint in 2.0% of the batches scheduled; the human scheduler relaxed the constraint in 1.3% of the batches. The minimum absolute level of relaxation was the same in both cases. In other words, the prototype relaxed this constraint slightly more often than the human scheduler, but the degree of relaxation was about the same. Considering that the human scheduler used an extra degree of freedom that the prototype did not use, the prototype's performance is in this regard better than the human scheduler's.

Execution Times: While the design goal of the prototype was schedule quality and not execution speed, the elapsed time of each test case was recorded to gain insight into the resources required by the research approach. The average total CPU time for the eight test cases was 504 seconds per schedule (interpreted Prolog running on a VAX 11/780). The actual elapsed time between the start and finish of schedule development was between 10 and 15 minutes, depending on the amount of output for sub-problem solutions and execution diagnostics, and the number of other users on the system.

CONCLUSIONS

The success of the prototype scheduling system suggests that an AI-based system might work effectively for the case study addressed by this research, and other industrial scheduling problems. However, the wealth

of prototype expert systems and scarcity of operational expert systems in industry suggests the use of caution. The primary concern is the extensibility of the approach to the objectives, constraints, and special cases not addressed by the prototype. To the extent that new requirements can be expressed as constraints on the space of feasible schedules, the prototype developed in this research can serve as the basis for an operational scheduling system. For the case study used in this research, the additional requirements needed to create an operational scheduling system could be expressed in this manner. The major limitation is the increase in execution time attendant with the addition of new constraints for evaluation.

Another concern is the maintainability of the system while in operation. The same software engineering practices used in a mathematical programming-based scheduling system would need to be applied in developing an operational AI-based scheduling system, because both must document how real-world constraints are expressed and used within the system.

The nature of the constraints found in the case study of this research permitted the decomposition of the problem along the time-line into sub-period problems. In addition, the initial decomposition of the problem into a hierarchy allowed the prototype to anticipate and reduce the possibility of interaction between sub-problems. Therefore, time-line decomposition should be used when the problem constraints create natural sub-divisions of time, and in conjunction with techniques that minimize the possibility of sub-problem interaction.

Within the framework of these considerations, the conclusions of this research can be summarized as follows:

- For parallel processor problems with a variety of objectives and constraints, including time-dependent constraints, AI methods can provide viable tools for developing computer-based scheduling and sequencing systems. For process operations similar to the case study used in this research, AI-based scheduling systems can develop schedules of a quality comparible to a human scheduler's for routine conditions.
- Hierarchical planning used in conjunction with constraint-directed search provides a framework for expressing both human scheduler's knowledge and analytic models of the physical system.
- Constraint satisfaction, as opposed to optimizing an objective function, is a useful alternative method for solving scheduling problems.
- Dividing the schedule period into sub-periods is a viable means for reducing problem complexity in scheduling problems like the case study used in this research. The use of sub-periods also provides a means to deal effectively with time-dependent constraints.

ACKNOWLEDGEMENTS

This research was funded in part by a Manufacturing Systems Engineering Intern Fellowship from E.I. DuPont de Nemours, Inc., in conjunction with the Department of Industrial Engineering and Operations Research at Virginia Tech, and by the Internal Research and Development program of The Charles Stark Draper Laboratory, Inc. The authors thank Draper Industrial Automation Division technical directors Rick Hildebrant, Ivan Johnson and Cynthia Whitney for their contributions to this analysis.

LIST OF REFERENCES

1. Reklaitis, G. V., "Review of Scheduling of Process Operations", Computer-Aided Process Design and Analysis, AIChE Symposium Series, Vol. 78, No. 214, 1982, pp. 119-133.

2. Seward, S. M., S. G. Taylor and S. F. Bolander, "Progress in Integrating and Optimizing Production Plans and Schedules", International Journal of Production Research, Vol. 23, No. 3, 1985, pp. 609-624.

3. Steffen, M. S. and T. J. Greene, "A Prototype System for Scheduling Parallel Processors Using Artificial Intelligence Methods", To appear in the IIE 1986 Spring Conference, Dallas, TX, May 11-14, 1986.

4. Bechler, E., J. Proth and K. Voyiatzis, "Artificial Memory in Production Management", Institut National de Recherche en Informatique et en Automatique, Research Report No. 336, Centre du Rocquencourt, France, September, 1984.

5. Bullers, W. I., S. F. Nof, and A. B. Whinston, "Artificial Intelligence in Manufacturing Planning and Control", AIIE Transactions, Vol. 12, No. 4, 1980, pp. 351-363.

6. Hall, M. D. and G. Putnam, "An Application of Expert Systems in FMS", AUTOFACT 6 Conference Proceedings, Society of Manufacturing Engineers, Dearborn, MI, 1984, pp. 2-26 to 2-39.

7. Kempf, K. G., "Manufacturing and Artificial Intelligence", Robotics, Vol. 1, No. 1, May 1985, pp. 13-26.

8. Mayer, R. J., R. E. Young, and D. Phillips, "Artificial Intelligence - Applications in Manufacturing", AUTOFACT 6 Conference Proceedings, Society of Manufacturing Engineers, Dearborn, MI, 1984, Proceedings Supplement.

9. McLean, C. R., "An Architecture for Intelligent Manufacturing Control", Proceedings of the 1985 ASME International Computers in Engineering Conference and Exhibition, Boston, MA, Aug. 4-8, 1985, pp. 391-397.

10. Shaw, M. J., and A. B. Whinston, "Automatic Planning and Flexible Scheduling: A Knowledge-Based Approach", Proceedings of the 1985 IEEE International Conference on Robotics and Automation, St. Louis, MO, March, 1985, pp. 890-894.

11. Tate, A., "Planning and Condition Monitoring in a FMS", International Conference on the Development of Flexible Automation Systems, London, England, July 10-12, 1984, pp. 62-69.

12. Tou, J. T., "Design of Expert Systems for Integrated Production Automation", Journal of Manufacturing Systems, Vol. 4, No. 2, 1985, pp. 147-156.

13. Villa, A., G. Murari and F. Lombardi, "An Expert Control System For Up-Dating Work Loads in a Flexible Manufacturing Cell", Proceedings of the 1985 ASME International Computers in Engineering Conference and Exhibition, Boston, MA, Aug. 4-8, 1985, pp. 457-462.

14. Young, R. E., "Planning and Control Requirements for Flexible Manufacturing Systems", Proceedings of the 1985 ASME International Computers in Engineering Conference and Exhibition, Boston, MA, Aug. 4-8, 1985, pp. 347-353.

15. Bourne, D. A. and M. S. Fox, "Autonomous Manufacturing: Automating the Job-Shop", IEEE Computer, Sept., 1984, pp. 76-86.

16. Fox, M. S., B. Allen and G. Strohm, "Job-Shop Scheduling: An Investigation in Constraint-directed Reasoning", Proceedings of the National Conference on Artificial Intelligence, Pittsburg, PA, Aug. 1982, pp. 155-158.

17. Fox, M. S., Constraint Directed Search: A Case Study of Job-Shop Scheduling, Doctoral Dissertation, Carnegie-Mellon University, 1983. Technical Report CMU-RU-TR-83-22.

18. Fox, M. S., S. F. Smith, B. P. Allen, G. A. Strohm and F. C. Wimberly, "ISIS: A Constraint-directed Reasoning Approach to Job Shop Scheduling", Proc. Trends and Applications 1983, National Bureau of Standards and IEEE Computer Society, May 25-26, 1983, pp. 76-81.

19. Fox, M. S. and S. F. Smith, "ISIS - A Knowledge Based System For Factory Scheduling", Expert Systems Journal, Vol. 1, No. 1, 1984, pp. 25-49.

20. Ow, P. S. and S. F. Smith, "Towards an Opportunistic Scheduling System", Proceedings of the Nineteenth Annual Hawaii International Conference on System Sciences, Honolulu, Hawaii, January 7-10, 1986.

21. Smith, S. F. and P. S. Ow, "The Use of Multiple Problem Decompositions in Time Constrained Planning Tasks", Proceedings of the 9th International Joint Conference on Artificial Intelligence, Los Angeles, CA, August, 1985, pp. 1013-1015. Carnegie-Mellon University Robotics Institute Technical Report CMU-RI-TR-85-11.

22. Project Sampler, Intelligent Systems Laboratory, The Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, May, 1985.

23. Papas, P. N., "ISIS - A Project in Review", National Bureau of Standards Symposium on Real-Time Optimization in Automated Manufacturing Facilities, Gaithersburg, MD, Jan. 21-22, 1986.

24. Fox, M. S. and S. F. Smith, Constraint-Based Scheduling in an Intelligent Logistics Support System: An Artificial Intelligence Approach, Annual Report, March 15, 1984 - March 14, 1985, Air Force Office of Scientific Research, Report No. AFOSR-TR-85-0720, July 15, 1985. Available through NTIS.

25. Evers, D. C., D. M. Smith and C. J. Staros, "Interfacing an Intelligent Decision-Maker to a Real-Time Control System", SPIE 1984 Applications of Artificial Intelligence, Society of Photo-Optical Instrumentation Engineers, Vol. 485, 1984, pp. 60-64.

26. Fox, B. R. and K. G. Kempf, "Opportuntistic Scheduling for Robotic Assembly", Proceedings of the 1985 IEEE International Conference on Robotics and Automation, St. Louis, MO, March, 1985, pp. 880-889.

27. Fukumori, K., "Fundamental Scheme For Train Scheduling", MIT AI Memo No. 596, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, Sept. 1980.

28. Glover, F., R. Glover and C. McMillan, "A Heuristic Programming Approach to the Employee Scheduling Problem and Some Thoughts on 'Managerial Robots", Proceedings of the Sixteenth Hawaii International Conference on System Sciences, Honolulu, Hawaii, January 5-7, 1983, pp. 420-436.

29. Goldstein, I. P. and R. B. Roberts, "NUDGE: A Knowledge-Based Scheduling Program", Proceedings of the Fifth International Joint Conference on Artificial Intelligence, Cambridge, MA, 1977, pp. 257-263.

30. Klar, W., "TABS - A Knowledge Based Time Planning System", International Conference on Networks and Electronic Office Systems, IERE, Reading, Berkshire, England, Sept. 26-30, 1983, pp. 171-175.

31. Orciuch, E. and J. Frost, "ISA: Intelligent Scheduling Assistant", Proceedings of the First Conference on Artificial Intelligence Applications, IEEE Computer Society, Dec. 1984, pp. 314-320.

32. Robbins, J. H., "PEPS - the Prototype Expert Priority Scheduler", AUTOFACT 7 Conference Proceedings, Society of Manufacturing Engineers, Dearborn, MI, 1985, pp. 13-10 to 13-34.

33. Vere, S., "Planning in Time: Windows and Durations for Activities and Goals", Pattern Analysis and Machine Intelligence, Vol. PAMI-5, No. 3, May 1983, pp. 246-266.

34. Cohen, P. R. and E. A. Feigenbaum, The Handbook of Artificial Intelligence, Vol. 3, William Kaufmann, Inc., Los Altos, CA, 1982.

35. Stefik, M. S., J. Aikins, R. Balzer, J. Benoit, L. Birnbaum, F. Hayes-Roth and E. Sacerdoti, "The Organization of Expert Systems: A Tutorial", Artificial Intelligence, Vol. 18, 1982, pp. 135-173.

36. Hayes-Roth, F., D. A. Waterman and D. B. Lenat (eds.), Building Expert Systems, Addison Wesley, Reading, MA, 1983.

37. Steffen, M. S., "An Application of Artificial Intelligence Methods to Scheduling Parallel Processors", Masters Thesis, Dept. of Industrial Engineering and Operations Research, Virginia Tech, Blacksburg, Va, August 1985.

38. Roach, J. W. and G. Fowler, "HC: The Virginia Tech Prolog/Lisp Manual", Dept. of Computer Science, Virginia Tech, 1983.

39. Clocksin, W. F. and C. S. Mellish, Programming in Prolog, Springer-Verlag, New York, 1981.

40. Lee, N. S. and J. W. Roach, "GUESS/1: A General Purpose Expert Systems Shell", Technical Report TR-85-3, Dept. of Computer Science, Virginia Tech, 1985.

41. Hax, A. C., "Aggregate Production Planning", Handbook of Operations Research Models and Applications, J. J. Moder and S. E. Elmaghraby (eds.), Van Nostrand Reinhold Co., New York, N.Y., 1978.

42. Stefik, M. S., Planning With Constraints, Doctoral Dissertation, Stanford University, 1980.

43. Stefik, M. S., "Planning With Constraints (MOLGEN: Part 1)", Artificial Intelligence, Vol. 16, 1981, pp. 111-140.

44. Steffen, M. S. and T. J. Greene, "An Application of Hierarchical Planning and Constraint-directed Search to Scheduling Parallel Processors", To appear in the 1986 IEEE International Conference on Robotics and Automation San Francisco, CA, April 7-10, 1986.

45. Sussman, G. J., A Computer Model of Human Skill Acquisition, American Elsevier, New York, 1975.

46. Hayes-Roth, B. and F. Hayes-Roth, "A Cognitive Model of Planning", Cognitive Science, Vol. 3, 1979, pp. 275-310.

A TWO-LEVEL PLANNING AND SCHEDULING APPROACH FOR COMPUTER INTEGRATED MANUFACTURING

Michael Shaw

University of Illinois at Urbana-Champaign

Introduction.

An emerging architecture for computer integrated manufacturing (C.I.M.) systems is the cellular system, as shown in Figure 1, consisting of flexible cells (Cutkosky 1984); each cell can communicate with other cells through a local area network (LAN). Such cellular manufacturing systems have played an increasingly important role in the design of the fully automated systems for many reasons; among them are reduced machine set-up time, reduced tooling, the simplification of planning and control, reduced in-process inventory, the near-constant load-time and system modularity (McLean 1983, Sikha 1984).

The scheduling method described in this paper takes into account the characteristics of local area network for communication. The system is treated as a loosely-coupled network of cooperating cells and the scheduling is carried out by a network-wide bidding scheme for determining the assignment of cells to given jobs dynamically. It is a distributed scheduling method in that no node in the network has greater importance, as far as scheduling is concerned, than any other node. Moreover, this scheduling method can incorporate different dispatching rules and can be used for both task allocation and resource allocation. As such, this is the only research in the manufacturing area to date that takes into account the use of local area networks for executing job scheduling and we will show that there are ample advantages for doing so.

II. A Two-level Scheduling Approach.

Associated with such a network environment, there are two possible control structures underlying the scheduling decisions:

1) to use a centralized scheduler in charge of job assignment. The scheduler keeps track of the whole cellular system by a global database.

2) to use a distributed scheduling scheme and let the set of cells perform scheduling based on local information.

By way of comparison, scheduling with distributed control has these advantages:

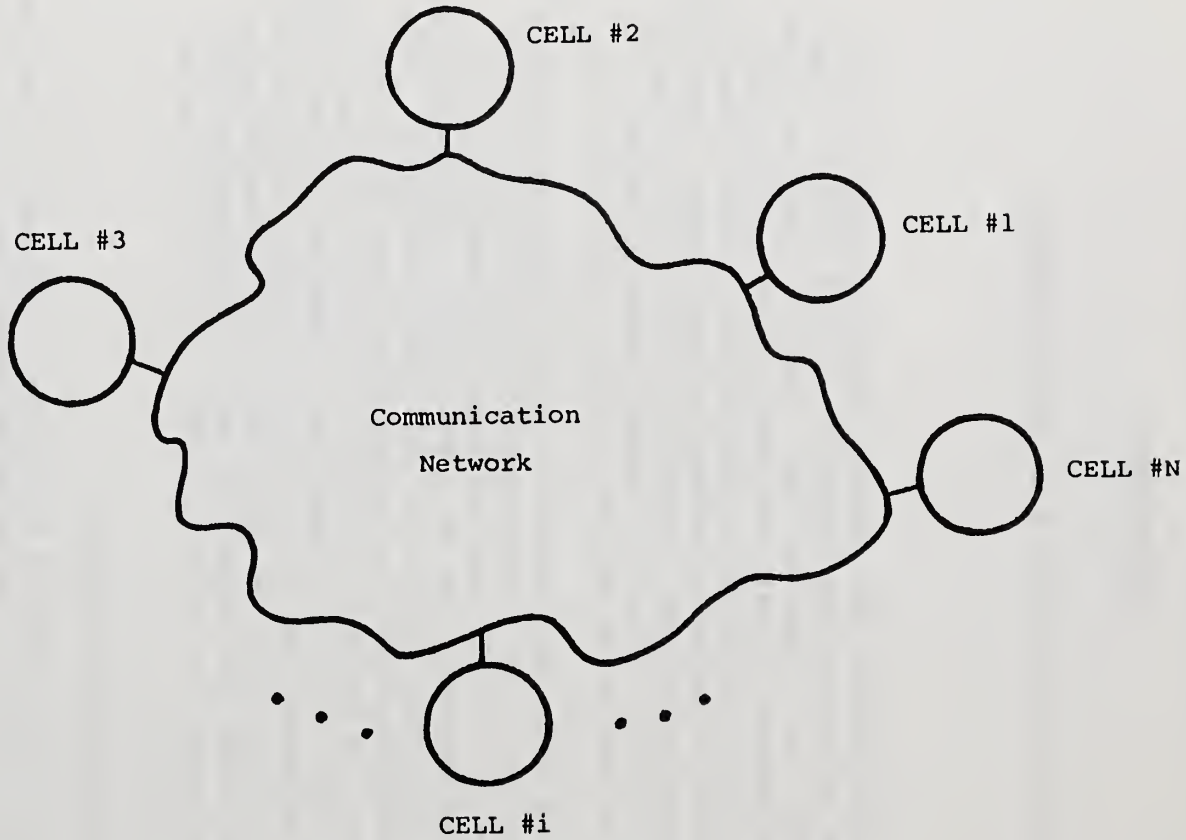


Figure 1. A Conceptual Model of the Cellular CIM

(1) better reliability--the system degrades gracefully in the face of scheduler breakdown, (2) upward extensibility--the control structure remains the same with additions of new cells to the extent that the network is not saturated, (3) improved performance—the scheduling performance can be improved because the scheduling is achieved by parallel processing and also because of the elimination of the bottleneck associated with the global scheduler, and (4) cost effectiveness—it is more cost effective because of the smaller processing requirements on the computers and less communication activities needed for global updating.

The adoption of distributed scheduling method implies the need for a new type of information control mechanism for coordinating manufacturing activities. Since there is no centralized master controller directing the activities of individual cells, it becomes essential that the cells have to be able to reach scheduling decisions by collective, concerted efforts. Two major issues warrant attention:
(1) an effective task allocation scheme among cells to ensure that all the resources can be efficiently utilized, and (2) the coordinating mechanism exercised among the cells, carrying out manufacturing tasks cooperatively. The network-wide bidding scheme described in this paper can achieve these two functions.

Such an approach essentially treats the scheduling by a multi-agent problem-solving paradigm: because the whole scheduling problem is too complicated, the set of problem-solving agents--the cells would carry out the tasks collectively. Just as in human organizations, bidding is employed as a mechanism for coordinating the execution of tasks among the cells. This paradigm was developed by research in artificial intelligence (Davis 1983, Shaw 1985) and has been applied to various types of distributed systems. As such, the distributed scheduling method can be viewed as consisting of two-level: The first-level scheduler dynamically assigns jobs to the most appropriate cells and the second-level scheduler executes scheduling within each cell (Figure 2). This two-level approach will be described in more detail in the next two sections.

III.1 The First Level: A Network-wide Distributed Scheme for Dynamic Task Assignment.

In the network-wide bidding scheme, when a cell needs to initiate the task assignment algorithm for one of its jobs, it begins with broadcasting a task-announcement message through the LAN to other cells and takes on the role as the manager cell of the job. Those cells that receive this message will, in turn, transmit a bidding message which contains its estimation of the earliest finish time, the surrogate for the "price" of the job if assigned. When all the bids have returned, the manager cell then selects the cell which can finish the job the earliest to perform the task. The corresponding workpiece is then transferred to the cell selected, or the contractor cell.

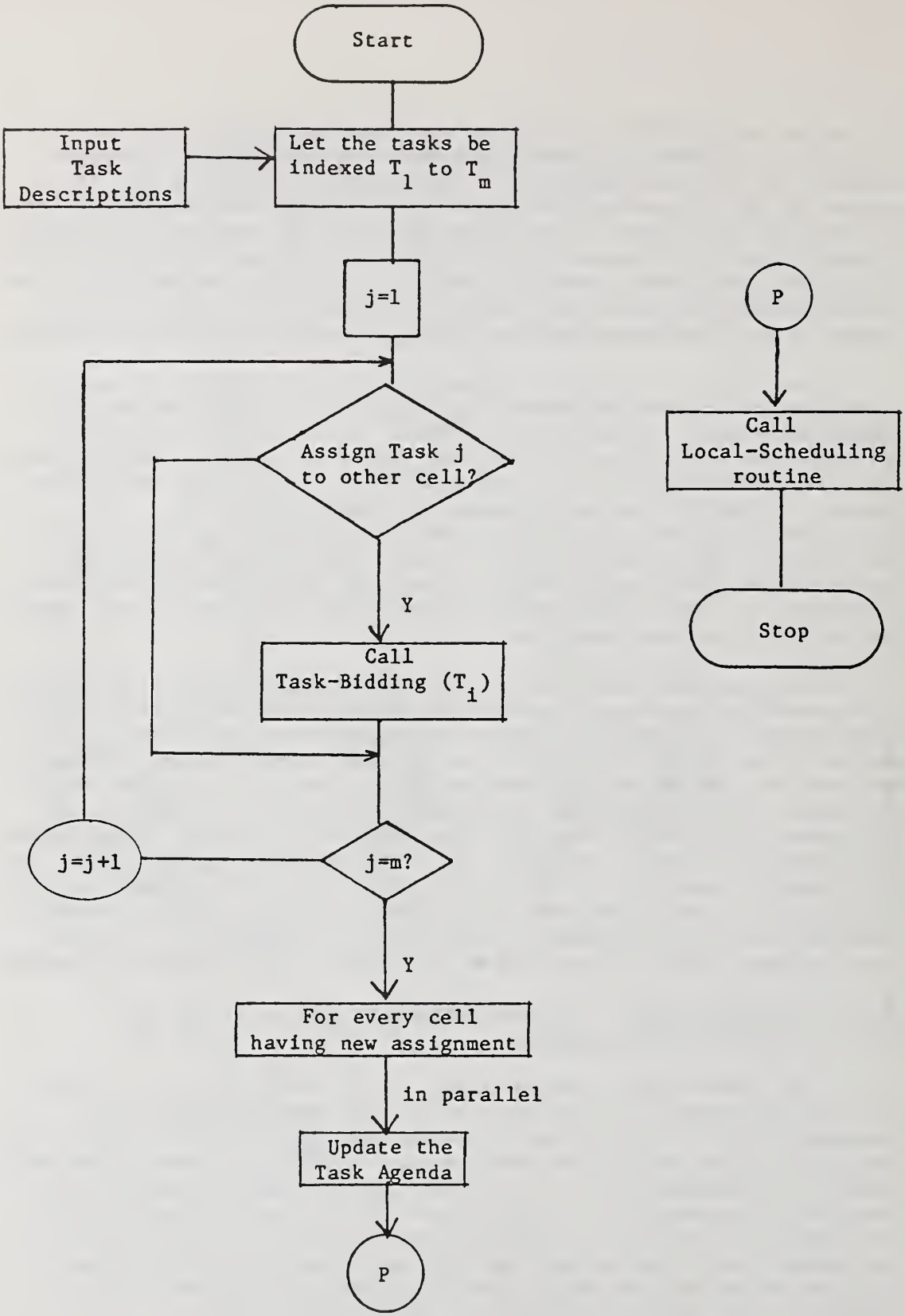


Figure 2. The Flowchart for the Two-Level Scheduling Approach

Task Announcement.

When a job finishes its operations in a cell, the cell's control unit will check to see if there are any remaining operations to be done. If all operations have been completed, the work piece is sent to the storage area; otherwise, the cell's control unit would have to make the decision regarding which cell the job should go to next. Keeping the job in the same cell is also a valid decision, but this has to be made after the performance data from other cells are collected and compared through bidding.

In the cellular manufacturing system, three types of manufacturing cells may exist: (1) flexible cells, where general purpose machines are used and the set-up is flexible for performing a wide ranging family of operations; (2) product-oriented cells, where a certain type of product is manufactured, e.g., a gear cell for producing gears; and (3) robot assembly cells, where robots are used for putting sub-assemblies together. Depending on the set up of a flexible cell or a robot assembly cell, the cell's control unit would give different performance estimates at different moments. The product-oriented cells on the other hand, have relatively more static functions in terms of the set of operations they perform. For a job requesting operations that can be performed in these product oriented cells, the task-announcement message can be directly addressed to the destination cell. The scheduling of jobs can be accelerated by such "focussed addressing."

Bidding

When a cell receives a task-announcement message from the communication network, it first matches the task description with its capability-list and checks to see if the required operations are within its capabilities. A bid for the task is returned only if the cell can perform the task. The cell then proceeds to calculate the bidding function which has the following three components. (1) The estimated processing time, which is calculated by a routine based on the machining parameters specified in the task-announcement packet, such as the cutting speed, the raw material, the depth of cut, surface finish requirements, the cutting tools wearing condition, the current setup, and the lubrication temperature; (2) the estimated waiting time, which is calculated by adding up the estimated processing time of the jobs in the queue; (3) the estimated travel time, which is calculated based on the travel distance between the two cells.

This particular bidding function implies that each flexible cell submits its estimation on the earliest time it can finish the task if assigned. By assigning the task to the lowest bidder, the manager cell essentially is executing the earliest-finishing-time (EFT) heuristic for dynamic scheduling (Baker 1974). Other dispatching heuristics can also be incorporated. For example, if the bidding function is determined by the estimated processing time of each cell, then the scheduling

is essentially based on the decentralized version of shortest-processing-time (SPT) dispatching, which has been shown to give good scheduling performance to dynamic job shop (French 1982) and flexible manufacturing systems (Chang 1984).

Bid Evaluation and Task Awarding.

When the deadline for bid submission is due, a bid-evaluation procedure is carried out by the cell that originally announced the task. All the bids submitted for this task have been put in a list, ranked by the value of each bid. In our algorithm, the bid of each cell i is calculated based on the earliest finish time of each task if the task is assigned to cell i. The scheduler of the manager cell then chooses the cell with the smallest bid, i.e., which can finish processing the task earliest.

Once bid-evaluation is completed, an award message is sent to the best bidder, informing the awardee of the pending job so that the cell which has been awarded the task will take this new task into consideration in the subsequent calculation of earliest-finishing-time in bidding for future jobs. This task-awarding information also enables the awardee cell to start loading part programs for the new task. The local scheduler of the awardee cell will take the newly assigned job into consideration in the next scheduling cycle. The bidding scheme is schematically shown in figure 3.

Under the distributed control scheme, the dynamic system information such as cell status, location of parts, position of tools, progress of jobs, etc., is managed by a distributed database system. Each cell maintains its own local world model, while systematically coordinating with other cells through task sharing and bidding. By eliminating the necessity to collect dynamically changing system information in a global database, the possible bottleneck and the communication activities for constant updating are avoided.

III.2 Evaluating the Distributed Scheduling Scheme: A Simulation Study.

To evaluate the performance of the network-wide bidding scheme as a dynamic scheduling algorithm, we have conducted a simulation study on hypothetic cellular flexible manufacturing systems. The primary objective of the simulation study is (1) to compare the performance of the bidding algorithm with other approaches used in prior scheduling research. Specifically, we compared the bidding algorithm with the dynamic dispatching method; and (2) to evaluate the performance of bidding algorithm with different bidding functions. For this purpose, the SPT heuristic and the EFT heuristic are evaluated.

In effect, the scheduling problem of the cellular system is partitioned into two decisions:

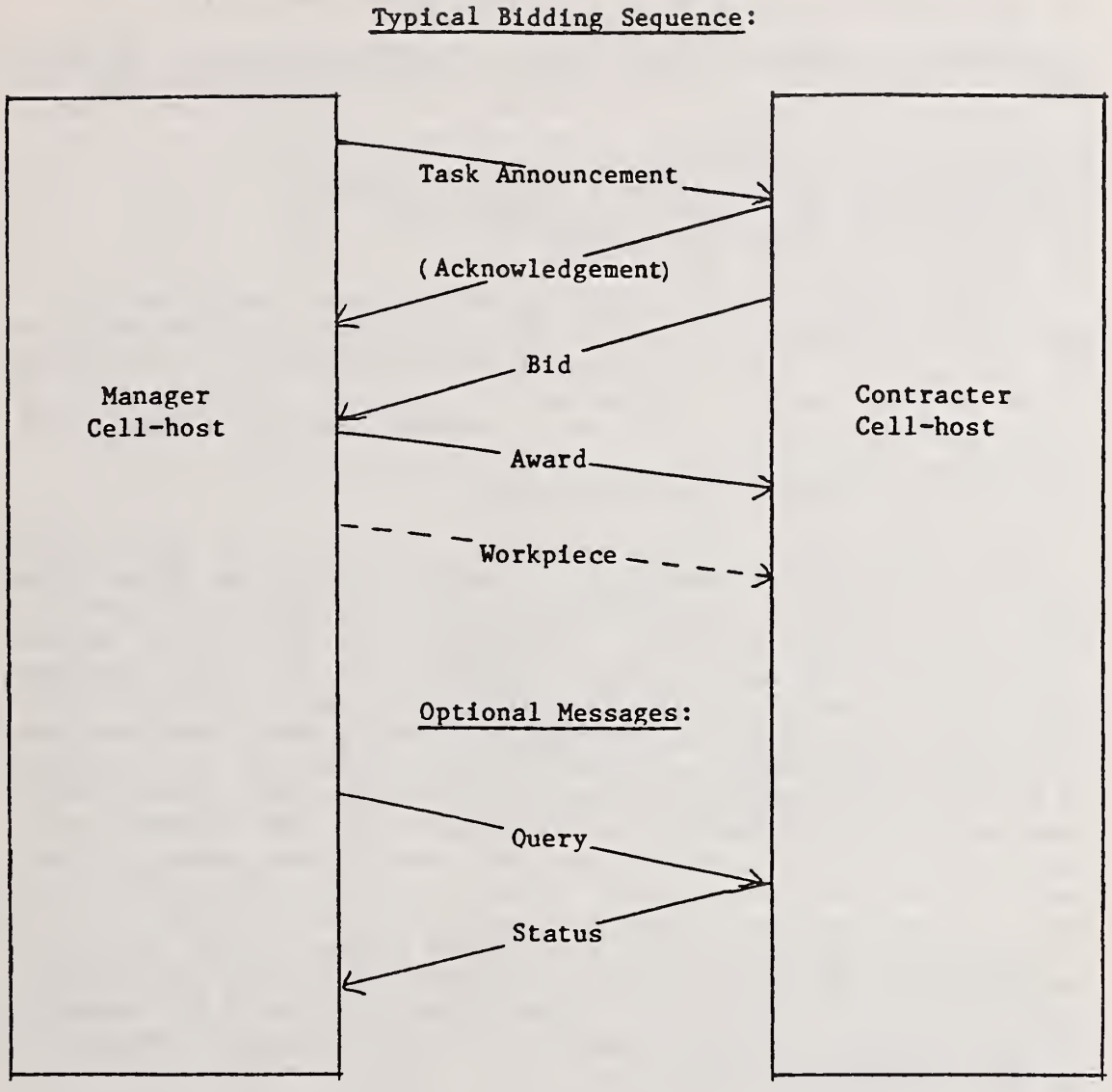


Figure 3. Information Flows in the Bidding Sequence

(1) the assignment of jobs to the appropriate manufacturing cells; and
(2) the sequencing and scheduling of jobs within each cell.

The response variables gathered from the simulation runs are the following:

(1) job flow-time statistics;
(2) proportion of jobs failing to meet the due date;
(3) job lateness and tardiness statistics; and
(4) average in-process waiting time.

As described in the objectives of the simulation study, we are especially interested in comparing the performance between bidding-SPT and bidding-EFT to evaluate the two scheduling heuristics incorporated in the bidding function. Furthermore, by comparing the performance of the bidding-SPT and myopic-SPT, we can evaluate the characteristics of distributed scheduling with the bidding mechanism against centralized scheduling with myopic dispatching rules.

Among the simulation performance data, two particular results stand out: (1) bidding-EFT clearly has the best performance in terms of mean flow-time, tardiness, and in-process waiting time measures. (2) In 10 out of the 12 simulation runs, the bidding-SPT method performs better than the myopic-SPT method, also in terms of mean flow-time, tardiness, and in-process waiting time. The distributed scheduling method performs better than the centralized counterpart primarily due to the fact that by executing the bidding mechanism, the scheduling decision is achieved by cells collectively based on purely local information stored within each cell. If the scheduling was to be done with centralized control, then there must be a global database and thereby large amount of communication activities are needed to keep the dynamic information up-to-date. In contrast, by letting each individual cell estimate its "price" for performing the announced tasks, all the estimation and calculation can be done based on information stored within the cell, and message-passing is carried out only to announce task or submit bid. Therefore, the distributed scheduling scheme utilizes more accurate information for estimating scheduling heuristics.

IV. The Second Level: A Nonlinear Planning System for Local Scheduling.

The second-level scheduler is a nonlinear planning system which schedules tasks within each manufacturing cell. The approach is based on the pattern-directed inference method with state-space searching, similar to the STRIPS system (Fikes and Nillson 1971). It is called nonlinear planning because the resulting plan is partially ordered. A prototype of this nonlinear planning system, referred to as XCELL, has been implemented on VAX 11/780 under the UNIX environment (Shaw 1986a). Written in LISP, XCELL consists of three components: declarative knowledge, domain specific knowledge and a control system.

The Declarative Knowledge.

Three kinds of manufacturing knowledge are stored in the database of XCELL: the world model which describes the working environment, the task descriptions, and the production plans; first-order predicate literals are used in the world model for knowledge representation. Also stored at the data level is the representation of the manufacturing plan by a partially ordered network of activities. The plan representation is used to monitor the execution of the planned activities; if there are deviations between the conditions specified by the plan and the conditions in the real world, XCELL should modify the rest of the plan by invoking a plan-revision routine. The plan representation can also be used to accommodate dynamically changing job-mix, which present to XCELL in the form of a goal structure.

The Domain-specific Knowledge.

A set of operators, stored in XCELL's knowledge base, is used to represent actions which the system may perform. Each operator contains information about the object that participates in the actions, what the actions are attempting to achieve, the effects of the actions when they are performed, and the necessary conditions for the actions to be performed. In addition to the standard AI-based planning formalism--which specifies an action by the add-list, delete list, and preconditions (Nilsson 1980)—XCELL included two more descriptions for each action--the "resource" used during the action, and the "duration" of the action. There are two advantages to this addition: the increased representational power of the action model and the resulting acceleration of conflict detection and conflict resolution in the decision process.

The Control System.

At the control level, an embedded inference engine is used to develop and organize the necessary actions to accomplish the goals. For a M-part-N-machine scheduling problem the goal can be naturally decomposed into M subgoals, with each subgoal generating a linear plan for the corresponding part. Based on the nonlinear planning approach, the control level of XCELL will construct plans by the four-step algorithm:

The Nonlinear Planning Algorithm.

Step 1: Generate a linearly-sequenced plan for each subgoal.

Step 2: Identify problematic interactions between the actions of parallel subplans.

Step 3: Synthesize the subplans; construct precedence constraints between the pairs of conflicting actions to avoid harmful interactions.

Step 4: Executing plan-revision for alternative resources.

XCELL is organized as hierarchical planner consisting of three levels: strategic level, planning level, and operational level. Because of the scheduling characteristics of the flexible manufacturing cell, the strategic level of XCELL can choose four different planning modes: static planning, dynamic planning, plan-revision, and simulation. The inference engine of XCELL can execute either forward-or backward-chaining to construct nonlinear plans. Currently we are conducting a computation study to evaluate the effectiveness of various algorithmic designs and heuristic search methods.

Conclusions.

We have shown a two-level method for dynamic scheduling in cellular flexible manufacturing systems. The method has the following features:

1) It is a distributed scheduling technique; no node has greater importance, as far as scheduling is concerned, than any other node.

2) The algorithm is flexible, and can take into account such information as loading factor, unexpected break-downs, or resource constraints in the bidding scheme.

3) Compared with dynamic dispatching rules previously used, the bidding algorithm is characterized by its more accurate estimation of processing times, without spending the cost of constant updating. The performance improvement by such information is verified by simulation results.

4) This is the only scheduling algorithm in the manufacturing area to date that considers the characteristics of the communication network, i.e., loosely coupled nodes with distributed control, packet-switching, communication delay, and the broadcasting capability.

REFERENCES.

Baker, K., 1974, Introduction to Sequencing and Scheduling (New York: John Wiley & Sons).

Chang, Y. L., Sullivan, R. S., Bagchi, U., 1984, Experimental Investigating of Quasi-Realtime Scheduling in Flexible Manufacturing, Proc. First ORSA/TIMS Conference on FMS.

Cutkosky, M., Fussell, P., and Milligan, R., 1984, Precision Flexible Machining Cells Within a Manufacturing System, Technical Report CMU-RI-TR-84-12, The Robotics Institute, Carnegie-Mellon University.

Davis, R. and Smith, R., 1983, Negotiation as a Metaphor for Distributed Problem Solving, Artificial Intelligence, Vol. 20, pp. 63-109.

Fikes, R. E. and Nilsson, N. J., 1971, STRIPS: a New Approach to the Application of Theorem Proving to Problem Solving, Artificial Intelligence, 2(3/4), pp. 189-208.

French, S., 1982, Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop (New York: John Wiley).

Nilsson, N., 1980, Principles of Artificial Intelligence, (Tioga: Palo Alto).

Shaw, M., 1986a, XCELL: A Nonlinear Planning System for FMC Scheduling and Planning, Working paper, Dept. of Bus. Adm., University of Illinois.

Shaw, M., 1986b, A Distributed Scheduling Method for Computer Integrated Manufacturing: The Use of Local Area Networks in Cellular Systems, Working paper, Dept. of Bus. Adm., University of Illinois.

Shaw, M. and Whinston, A., 1985a, Automatic Planning and Flexible Scheduling, Proc. Int. Conference on Automation and Robotics, (St. Louis, MO).

Shaw, M. and Whinston, A., 1985b, Task Bidding, Distributed Planning, and Flexible Manufacturing, Proc. of IEEE Conference on Artificial Intelligence Applications (Miami, FL).

MATCH-UP REAL-TIME SCHEDULING*

James C. Bean
John R. Birge

Department of Industrial and Operations Engineering
The University of Michigan
Ann Arbor, Michigan 48109

1. Introduction

A vast majority of the literature dealing with production scheduling involves determining a good schedule over a substantial time frame assuming that all problem characteristics are known. Such schedules have two uses. As a planning tool, they can be used to order material, set work schedules and other functions preliminary to the actual production. As an operational device, they can be used to direct step–by–step production operations.

In this second capacity such deterministic scheduling procedures typically run into difficulty. Once the production process begins, random disruptions can force the system out of the prescribed states rendering the preformulated schedule invalid. Such disruptions can include delays in the arrival of materials or components, quality rejection of material or components, machine breakdown or operator absences.

We would like to anticipate such disruptions during the pre–scheduling of the system and build a schedule with recourse for each contingency. Research such as Pinedo and Ross [1980], Glazebrook [1981], and Pinedo [1983] is currently improving capabilities in these directions. At this time, the available techniques are not able to solve problems of the size and complexity needed to make operational contributions to actual production systems.

In the absence of a tractable optimal technique most scheduling practitioners use control type real–time heuristics. Most common are priority list scheduling rules. In such techniques available jobs are ordered by some simple rule. When a machine becomes available, the job heading the list is started. These techniques are myopic and can lead to substantial error.

This paper develops the framework for an alternative approach to the disruption problem that yields operational heuristic scheduling systems. In this approach we seek to use the information captured in the deterministic pre–schedule, while altering its details to compensate for disruptions. Such alterations must be done in real time as the disruptions occur. In order to react in real time while retaining the "goodness" of the schedule, we do not completely reschedule tasks, but rather, adapt the old schedule to smooth out the difficulties created by the disruption and match–up with the pre–schedule. This approach has intuitive appeal since material flows have been set according to this pre–schedule. We show that the optimal schedule, given the disruption, attempts to match

* This material is based upon work supported by the National Science Foundation under Grants Nos. ECS–8304065 and ECS–8409682.

up to the pre–schedule. Hence, by attempting to match–up in our heuristic we are moving in the appropriate direction.

In this paper we propose a dynamic optimization formulation of a general scheduling problem based on problems faced at a large auto manufacturer. This allows the use of economic turnpike theory (see McKenzie [1976]) as a foundation for adaptive approaches to real–time scheduling. Turnpike results are asymptotic. In other words, adaptive approaches are justified in the limit as the horizon of the problem is increased. Implementation of match–up procedures requires choosing a finite time where the schedule is expected to match up with the predetermined schedule. The imposition of this finite match–up time may cause error. However, bounds may be developed for these errors.

Section 2 includes a formal model of the problem class being addressed and the procedure proposed here. Section 3 contains a theoretical justification of the procedure based on turnpike theory. Sections 4 and 5 include a discussion of the implementation of adaptive scheduling and a derivation of error bounds. Finally, Section 6 contains a summary and conclusions.

2. Problem Statement

Though most formulations of scheduling problems deal with a finite set of jobs, implying a finite problem horizon, actual scheduling problems have indefinite horizons. Though a finite set of jobs is known at any point in time, as those jobs are completed new jobs are defined so that the process continues indefinitely. The assumption that we may truncate the problem to currently known jobs may not be valid due to the introduction of end–of–study effects (see Baker [1977]).

In this section we view the scheduling problem as a deterministic infinite horizon optimization problem. Over the infinite horizon an infinite number of jobs will be defined and completed, but we assume that at any finite time there are only a finite number of jobs to consider. Let n_k be the maximum number of jobs at any finite time k.

The scheduling system is modeled as a sequence of decisions detailing task processing during the next time period. The system can be viewed as being in one of many states, each of which details the accumulated processing of known jobs. Resource assignments may also be included in state definition without difficulty but are omitted here to simplify notation. Constraints on the system such as task precedence, preemptability, and resource capacities can be seen as restrictions on the transition of the system from one state to another.

2.1 Notation

For completeness a list of all notation used in the paper is included here. The need for some of the following will become clear later in the paper.

$n_k \in Z$: The number of jobs in the system at time k.

$N \in Z$: A finite horizon time.

$x_k \in \Re^{(n_k)}$: The state of the system at the end of processing in period k. The i^{th} element of x_k, x_k^i, is the accumulated processing of job i.

x_k^*: The state passed through at time k by an undisrupted weakly optimal schedule.

X_k^*: The set of states corresponding to any undisrupted weakly optimal schedule.

$x_k^*(\bar{x})$: The state passed through at time k by a weakly optimal schedule that begins at state $\bar{x}$ at time 0.

$\pi_k \in [0,1]^{n_k}$: A decision vector indicating the fraction of time each job is processed during period k.

$\pi \in \times_{k=1}^{\infty}[0,1]^{n_k}$: Decision vectors describing a schedule covering the infinite horizon.

π^*: A weakly optimal schedule.

$\pi^*(\bar{x})$: A weakly optimal schedule given that the system is in state $\bar{x}$ at time 0.

Π: The set of all feasible schedules, π.

$\phi_k(\pi) \in \Re$: The incremental cost from schedule π during period k.

$\Phi(\pi; N) \in \Re$: The cumulative cost of schedule π over the first N periods, $= \sum_{k=1}^{N} \phi_k(\pi)$.

$\Phi(\pi) \in \Re$: The cumulative infinite horizon cost $= \lim_{N\to\infty} \Phi(\pi; N)$.

$\Phi'(a, b) \in \Re$: The minimal cost of moving from state a to state b, where the periods for the states are not specified.

$\Phi^*(N) \in \Re$: The optimal cost over the first N periods, where any disruption restrictions are included.

$\hat{\Phi}^* \in \Re$: The optimal infinite horizon cost after a single disruption.

$\Phi^* \in \Re$: The optimal infinite horizon cost from $k = 0$ including disruptions.

$\bar{\Phi}^* \in \Re$: The optimal expected infinite horizon cost from $k = 0$ including disruptions.

$D_k \in \Re^{(n_{k-1})} \times \Re^{(n_k)}$: The set of feasible transitions at time k. That is, given the current state, x_{k-1}, and a potential next state, x_k, then $(x_{k-1}, x_k) \in D_k$ if and only if (x_{k-1}, x_k) represents a feasible transition. The set of feasible π_k, given the current state, can be inferred from D_k.

p_i: The processing time required for job i.

w_i: The weight assigned job i (to be used for weighted tardiness or weighted flowtime schedules).

$\bar{k}_i$: The time for matching up after the ith disruption.

$\underline{k}_i$: The time of the ith disruption.

$\bar{l}$: The time between a disruption and a match-up.

δ: The cost of following the match-up heuristic from a single disruption until return to the pre-schedule.

δ_i: The cost of following the match-up heuristic from $\underline{k}_i$ to $\bar{k}_i$.

d: The minimum cost from a single disruption state to any potentially optimal state at $\bar{l}$.

P_k: The set of potentially optimal states at period k.

d_i^1: The minimum cost from any state in $P_{\underline{k}_i}$ to any state in $P_{\bar{k}_i}$.

d_i^2: The minimum cost from any state in $P_{\bar{k}_i}$ to any state in $P_{\underline{k}_{i+1}}$.

2.2 Assumptions

A1) Total costs are the sum of incremental costs in each period.

A2) The incremental costs in each period are convex in the processing decisions.

A3) The objective is to minimize total costs.

A4) In weighted flowtime and tardiness problems there is a positive lower bound on w_i.

A5) There is sufficient slack time in the infinite horizon schedule that for any feasible initial state, $\bar{x}$, and schedule, π, there exists a feasible schedule, $\pi(\bar{x})$, and time $\bar{l}$, uniform over $\bar{x}$, such that for $k \geq \bar{l}$, $\pi_k = \pi_k(\bar{x})$. Further, the absolute cost to achieve this matching up, δ, is bounded above by U, uniform in $\bar{x}$. That is, any feasible schedule is reachable from any feasible initial state at reasonable cost.

A6) Several technical regularity conditions which are discussed in the appendix.

2.3 Infinite Horizon Objective

In infinite horizon optimization problems there are many ways to define the infinite horizon objective. If the infinite horizon costs converge, the optimal schedule is that with least cost. However, such convergence can usually be assumed only under sufficient discounting. Since scheduling problems are rarely discounted, all schedules will possibly have infinite cost.

Several definitions of optimality have been suggested for this case, including: average optimality (see Derman [1966]), 1–optimality (see Blackwell [1962]), overtaking optimality (see McKenzie), catching-up optimality (see McKenzie), forecast horizon optimality, and periodic forecast horizon optimality (see Hopp, Bean and Smith [1984]). To conform with the concepts of McKenzie, we use the concept of a weakly optimal schedule in the first part of the paper. As will be proven, weakly optimal implies average optimal.

We say that a schedule, π, *overtakes* another schedule, $\bar{\pi}$, if

$$\liminf_{N\to\infty} \sum_{k=1}^{N} [\phi_k(\pi) - \phi_k(\bar{\pi})] \geq \epsilon$$

for some $\epsilon > 0$. A schedule is *weakly optimal* if no other schedule overtakes it. We assume here that such a schedule exists. Given such a schedule, redefine $\phi_k(\pi)$ by subtracting the incremental cost of the weakly optimal schedule. Then, without loss of generality, the optimal cost is zero. We make the additional assumption:

A7) A weakly optimal schedule exists and its incremental costs have been used to normalize the cost function such that the infinite horizon optimal undisrupted cost is zero over all time intervals.

A schedule, $\hat{\pi}$, is said to be *average optimal* if

$$\liminf_{N\to\infty} \{\Phi(\hat{\pi}, N)/N - \Phi(\pi, N)/N\} \geq 0, \quad \forall \pi \in \Pi.$$

Lemma 1: *If π^* is weakly optimal then it is average optimal.*

Proof: In the definition of weakly optimal divide both sides by N. The result follows.■

The term *cost* used here represents any of several of the common scheduling objectives, as well as any more general objective that satisfies the stated assumptions. The objective primarily used for examples in this paper is weighted tardiness. Assumption A2 requires that the incremental costs be convex. This assumption is satisfied for most common scheduling objectives including weighted flow time and weighted tardiness.

Lemma 2: *If $w_i \geq 0$ for all i, then incremental weighted tardiness in period k is convex in processing time.*

Proof: As the processing for a particular job is increased, the tardiness of the job (if any) decreases linearly with slope $-w_i$ until there is no further tardiness. At that time it remains at zero regardless of further processing. The resulting curve is convex in processing of job i.■

It is equally simple to show that weighted flowtime is convex in processing for non–negative weights. Note that since the sum of convex functions is convex, the accumulated cost through time N is also convex for either of these criteria.

2.4 Feasibility

All feasibility conditions are assumed described by the sets of conditionally feasible decisions, D_k. Common constraints include non–preemptability, precedence, resource capacities, and maximum allowable tardiness. Each of these can be represented by restrictions which retain the necessary characteristics of D_k. For example, if a job is non–preemptable and its current processing is strictly between zero and its total processing time, the only feasible choice is to process until processing is completed. For precedence, processing a subsequent job would require that total processing on its predecessor equals its necessary processing.

3. Turnpike Results

The model developed in Section 2 can be viewed as an investment model similar to that described in Ramsey [1928]. McKenzie shows that under certain assumptions such models behave in a manner analogous to the matching–up described here. In the literature of economics and infinite horizon optimization these results are called turnpike theorems. The analogy is from shortest path problems on a surface road and turnpike network.

Clearly, the shortest path from one point on a turnpike to another on the same turnpike is that turnpike. Even if you live off the turnpike, if the destination is distant enough, the shortest path is to take surface streets to the turnpike and to take the turnpike from there. In our model the turnpike is analogous to the predetermined schedule (see Figure 1). The distance from the current state to the turnpike exists because some disruption has transferred the problem to an alternate state.

If these results can be shown to apply to a general form of the scheduling problem they have significant implications for appropriate heuristic approaches. After a disruption, the optimal new schedule would then converge to the optimal schedule derived before processing began. This pre–schedule is then a surrogate for the complex objectives involved in scheduling. If our heuristic is designed to return to the original schedule, we know that this is in fact the direction followed by the optimal schedule to the disrupted problem.

The strongest turnpike results require uniform convexity of the incremental costs. Though most measures used in scheduling problems do not satisfy this assumption, the stronger results are valuable for perspective and potential application to other objectives.

3.1 The Uniform Convexity Case

If the incremental cost functions are uniformly convex, as the system proceeds in time from the boundary condition (disruption), the sequence of states passed through by the system, and hence the decisions made, grow increasingly close to the states defined by the pre-schedule. The effect of the disruption diminishes.

The hypothesis of this theorem requires the concept of a uniformly convex function. Essentially, to be uniformly convex a function must be strictly convex and must not asymptotically approach a line. For a rigorous definition of this notion see McKenzie.

Theorem 3: *Given the assumptions of this paper, if $\phi_k(\pi)$ is uniformly convex for all k and π corresponding to D_k, then $x_k^*(\bar{x}) \rightarrow x_k^*$ as $k \rightarrow \infty$.*

Proof: This problem seeks to minimize the sum of uniformly convex functions. This is equivalent to maximizing the negatives of these functions. The negatives of uniformly convex functions are uniformly concave. The sum retains this characteristic. Hence, the conditions of Theorem 3 of McKenzie are satisfied. The conclusion of this theorem follows immediately.∎

The proof in McKenzie follows this line. The cost of passing through states x_k^* is zero by Assumption A7. There exists a feasible path from $\bar{x}$ which matches up with this schedule. By Assumption A5 this may be done by time $\bar{l}$ at a cost of no more than U. Hence, the cost of passing through $\{x_k^*(\bar{x})\}$ given an initial state of $\bar{x}$ must be better that U. By uniform convexity, if $\bar{x}_k^*$ is bounded away from x_k^*, then a uniform penalty will be charged each period. This leads to infinite cost and a contradiction. For further details see McKenzie.

Corollary 4: *Under the conditions of Theorem 3, $\pi_k^*(\bar{x}) \rightarrow \pi_k^*$ as $k \rightarrow \infty$.*

Proof: In general, $\pi_k = x_k - x_{k-1}$. The result follows immediately from Theorem 3.

3.2 The Case of Multiple Optima

When the incremental cost functions are not uniformly convex, the problem may have multiple optima. This is the case in most scheduling objectives, including weighted flowtime and tardiness. If there are multiple optima, different initial states could lead to weakly optimal schedules which converge to different turnpikes, each being optimal. The claim can still be made that as $k \rightarrow \infty$, the state of the disrupted system will approach the set of weakly optimal schedules.

Definition: $\rho(x_k, X_k^*) = \inf_{y \in X_k^*} ||x_k - y||$.

Theorem 5: *Under the assumptions of this paper,* $\lim_{k \rightarrow \infty} \rho(x_k, X_k^*) = 0$.

Proof: Follows directly from Section 7 of McKenzie.

3.3 Example

Consider the following single machine total tardiness problem. At the beginning of each seven day cycle four jobs become ready. Their total processing is six days. Their due dates are such that they can be completed without tardiness—provided there are no disruptions in the system. The optimal schedule over the infinite horizon is to sequence the jobs 1–2–3–4 each week. This schedule is weakly optimal as discussed above.

Assume that the machine is now down for the first four days of the upcoming week. This disruption will not only affect the schedule for this week, but for upcoming weeks as well. The consequence of the theorem above is that we know that the effects of this disruption will fade out and that the weekly schedule will approach the original weakly optimal schedule.

Data for the example appear in Table 1. Figure 2 displays the undisrupted, weakly optimal schedule and the optimal schedule given the disruptions. Note that after four weeks the two schedules are identical for all future periods.

4. Implementation

Theorems 3 and 5 cannot be used directly to control a scheduling system in real time. They do, however, suggest that the following heuristic may be efficient. Assume that a schedule has been determined and used to set material flows. When a disruption occurs, the pre–schedule becomes obsolete. Rather than rescheduling in real time using the original objective, substitute as the objective a desire to return at minimal cost to the pre–schedule at some future date. The pre–schedule is assumed to incorporate all important goodness characteristics. It should contain more of these characteristics than could be incorporated explicitly in real time. From Theorems 3 and 5 we know that this heuristic schedule is tracking in the same direction as the optimal schedule, given the initial state.

The implemented process contains the following steps.

Match–Up Heuristic

Step 0: Construct a pre–schedule.

When a disruption occurs

Step 1: Determine some future time, $\bar{l}$, where the pre–schedule is reachable from the current state.

Step 2: Reschedule from current time 0 to $\bar{l}$ beginning in the current state and ending in state $x_{\bar{l}}$ at time $\bar{l}$. Stop.

This heuristic does not guarantee an optimal solution in all situations. However, by Theorems 3 and 5 we know that $\bar{l}$ can be chosen large enough to get arbitrarily close to an optimal schedule. It is not, however, generally known how large $\bar{l}$ must be or when disruptions will occur. A large $\bar{l}$ may also lead to computational difficulties, making the method impractical. Given these potential problems, the algorithm must be implemented as a heuristic and $\bar{l}$ must be determined to balance error and effort. Methods for bounding the error are given in the following section.

5. Error Bounds

The problem formulation presented here represents too general a class of scheduling problems to derive tight bounds on the error caused by this class of heuristics. For a specific problem, bounds should be derived to help set $\bar{l}$. In the following sections we present a rough bound for the general problem and apply it to an example.

5.1 Single Disruption

We obtain bounds on the optimal schedule cost by comparing our heuristic with a myopic solution strategy. Sharper bounds exploiting problem structure may be obtained using the aggregation

procedure in Bean, Birge, and Smith [1984].

For a single disruption, we assume that the match-up heuristic for some disruption state returns to state $x_{\bar{l}}^*$. The cost of returning to $x_{\bar{l}}^*$ is assumed to be δ.

Definition: The set of *potentially optimal* states at some time k, P_k, is defined as the subset of states, feasible after the disruption, which may be on an optimal path. In the absence of information, this can be taken to be the set of states feasible after the disruption. Note that this involves two reductions: elimination of states no longer feasible, and elimination of states that can be proven not to be on any optimal path.

Let the optimal cost of continuation after disruption be $\hat{\Phi}^*$. A myopic solution strategy is to find a path with minimum cost d from the disruption state to any state in $P_{\bar{l}}$.

Lemma 5: *Given the assumptions of this paper, the single disruption optimal schedule cost* $\hat{\Phi}^*$ *is bounded by*

$$d \leq \hat{\Phi}^* \leq \delta.$$

Proof: An optimal path must pass from the disruption state to some potentially optimal state at $\bar{l}$. The value d is the minimum among these distances, giving the lower bound. The upper bound follows from Assumption A7.■

5.2 Multiple Disruptions

In general, the system will encounter more than one disruption. For this possibility, we again use a myopic approach and the match-up heuristic to obtain bounds on the optimal solution cost. We let $\bar{l}$ be the time between the ith disruption at $\underline{k}_i$ and the ith match-up time $\bar{k}_i$ (i.e., $\bar{k}_i = \underline{k}_i + \bar{l}$) for all i. We assume here :

A8) If consecutive disruptions occur at times $\underline{k}_i$ and $\underline{k}_{i+1}$, then all potentially optimal states at $\underline{k}_{i+1}$ are reachable from all potentially optimal states at $\underline{k}_i$.

A9) $\underline{k}_{i+1} \geq \bar{k}_i$ for all i. That is, disruptions are sufficiently infrequent that we can match-up before the next disruption.

These assumptions ensure the feasibility of the match-up heuristic.

Let M be the number of disruptions for some finite horizon problem. The myopic approach is to find the minimum cost schedule path from $P_{\underline{k}_i}$ to $P_{\bar{k}_i}$ and from $P_{\bar{k}_i}$ to $P_{\underline{k}_{i+1}}$. Following this idea leads to a local minimization for lower bounds. Let

$$d_i^1 = \min\{\Phi'(a,b) : a \in P_{\underline{k}_i}, b \in P_{\bar{k}_i}\}, i = 1, \ldots, M,$$

$$d_0^2 = \min\{\Phi'(x_0, b) : b \in P_{\underline{k}_1}\},$$

and

$$d_i^2 = \min\{\Phi'(a,b) : a \in P_{\bar{k}_i}, b \in P_{\underline{k}_{i+1}}\}, i = 1, \ldots, M-1,$$

where $\Phi'(a,b)$ is the minimum cost over all paths connecting states a and b. These values are lower bounds on the values that the myopic approach could obtain. They also bound the cost of any other path over their defined ranges.

The match-up heuristic finds the minimum cost from the state following the disruption to $x^{\cdot}_{k_i}$ (within the disruption restricted set of feasible paths). Let this cost be δ_i. All other costs are zero for the match-up heuristic by Assumption A7. These observations are contained in the following theorem.

Theorem 6: *Given the assumptions of this paper, the multiple disruption optimal cost Φ^* is bounded by*

$$\sum_{i=1}^{M}(d_i^1 + d_{i-1}^2) \leq \Phi^* \leq \sum_{i=1}^{M}\delta_i,$$

where there are M disruptions.

5.3 Random Disruptions

All of the results above refer to a deterministic environment where all disruptions are known in advance. Instead, we would like to allow for a distribution on the disruptions and then find the optimal expected cost solution. For this stochastic environment, we define a probability space $(\Omega, \mathcal{B}, P)$. Each sample point ω corresponds to a scenario of occurrence times and durations of disruptions. All deterministic quantities defined above are interpreted as random variables when they are indexed by ω. This allows us to consider the stochastic case without defining extensive additional notation. The quantity $\Phi(\bar{\pi},\omega)$ then denotes the infinite horizon cost of following schedule strategy $\bar{\pi}$ under the conditions corresponding to ω. We wish to find

$$\bar{\Phi}^* = \min_{\pi} \mathrm{E}(\Phi(\pi,\omega)), \tag{1}$$

where $\mathrm{E}(\cdot)$ denotes mathematical expectation. We again assume that the disruptions are distributed so that Assumptions A8 and A9 hold almost surely.

Lemma 7: *Given the assumptions of this paper, the optimal expected cost $\bar{\Phi}^*$ is bounded by*

$$\bar{\Phi}^* \geq \mathrm{E}(\Phi^*(\omega)) \geq \mathrm{E}(\sum_{i=1}^{M(\omega)}(d_i^1(\omega) + d_{i-1}^2(\omega)).$$

Proof: Let $\bar{\pi}^*$ solve (1), then $\Phi^*(\omega) \leq \Phi^*(\bar{\pi}^*,\omega)$. Integration yields the first inequality. The second inequality follows from Theorem 6. ∎

Lemma 7 leads to the following result.

Theorem 8: *Given the assumptions of this paper, the optimal expected cost $\bar{\Phi}^*$ is bounded by*

$$\mathrm{E}(\sum_{i=1}^{M(\omega)}\delta_i(\omega)) \geq \bar{\Phi}^* \geq \mathrm{E}(\sum_{i=1}^{M(\omega)}(d_i^1(\omega) + d_{i-1}^2(\omega)). \tag{2}$$

Proof: The left-hand side of the expression is the expected value of following the match-up strategy. Since it is a feasible strategy, its expected value is an upper bound on the optimal expected cost. The second inequality follows from Lemma 7.∎

The use of these bounds is best demonstrated in an example. In this example, we make assumptions that actually reduce the interval in (2) to a point. For the infinite horizon objective we use expected average cost. In this case, if $\bar{\Phi}^*(N)$ is the optimal expected cumulative disruption cost for an N period horizon, then

$$\bar{\Phi}^*_{ac} = \lim_{N\to\infty} \frac{\bar{\Phi}^*(N)}{N}$$

Assuming each disruption causes some loss that is finitely positive, an infinite number of disruptions cause an infinite amount of loss. For this section the weakly optimal definition of optimality is not sufficient. Hence, we use average optimality. This gives us the average loss per period due to disruptions. The single disruption results of Theorems 3 and 5 still apply for this objective since all weakly optimal solutions also minimize average cost (Lemma 1). Matching up with the pre–schedule at some point must, therefore, lead to a minimum average cost solution. The bound in (2) can be applied for any finite horizon N as well as in the limit so that (2) can be used to bound $\bar{\Phi}^*_{ac}$.

The example includes two jobs which are cyclically available and due. Each job has a weight of one. Let Job 1 be ready at times $5t$, have a processing time of three, and be due at times $5t+3$, for $t = 0, 1, 2\ldots$. Let Job 2 be ready at $5t$, have a processing time of one, and be due at $5t+5$. Then the optimal pre–schedule processes Job 1 at $5t$ and Job 2 at $5t+3$ leaving one unit of slack in each production interval. We assume that disruptions occur with equal probability at 15, 16, 17, 18 or 19 time units since the last disruption and that disruptions last one or two time units with independent equal probabilities.

We choose a match–up time $\bar{l} = 10$ and assume that jobs are resumed after disruption with no loss of processing time. Note that the potentially optimal states at k are $\{x^*_k\}$ for $k(\text{mod } 5) = 0$, 1, 2 or 3 and $x_k = x^*_k$ or $x_k = (3,0)$ for $k(\text{mod } 5) = 4$. In all cases, $\min\{\Phi'(a,b) : a \in P_{\underline{k}_i}, b \in P_{\bar{k}_{i+1}}\} = \Phi'(x^*_{\underline{k}_i}, x^*_{\bar{k}_{i+1}})$. Note also that all costs are non–negative. These two observations imply that, for this problem,

$$\delta_i(\omega) = d^1_i(\omega),$$

and

$$d^2_i(\omega) = 0.$$

From this we obtain

$$\bar{\Phi}^*_N = \mathrm{E}\Big(\sum_{i=1}^{M_N(\omega)} \delta_i(\omega)\Big),$$

and

$$\bar{\Phi}^*_{ac} = \mathrm{E}(\delta_i(\omega)/N_C(\omega)),$$

where M_N is the number of disruptions in the first N periods and N_C is the length of a cycle between disruptions.

For this example the match–up solution produces an optimal average cost infinite horizon solution. The minimum expected average cost can then be calculated by weighting the costs over all cycles. The cycles are found by observing that disruptions occur at 0, 1, 2, 3 or 4 units into a

production interval with equal probabilities. Tardiness to the match–up point is then found and weighted by the inverse of the expected time to the next disruption. The resulting values appear in Table 2.

Cyclic availabilities and requirements for jobs are reasonable in many circumstances, but the distribution assumption of this example is restrictive and would not be met in general. The assumptions are however made to show the utility of the bounds. In specific applications, problem structure could be used to generate bounds.

In practice, schedulers can use the bounds in (2) to determine the time to match–up $\bar{l}$. They can consider the possible disruption patterns and calculate recovery costs under different scenarios to find the interval in (2). They can then vary $\bar{l}$ so that this interval and the computational burden for matching up are properly balanced.

6. Summary and Conclusions

Extensive deterministic pre–schedules are often inadequate to operate large systems due to unforeseen system disruptions such as machine failure. A common solution is real–time list processing control type algorithms. We present an alternative here that exploits the global attributes of the pre–schedule. It then adapts by matching up with the pre–schedule after disruption.

We have implemented this match–up heuristic in a computer program developed for a large auto manufacturer. The program considers a system of parallel nonidentical machines. Jobs have ready times and due dates. Each job requires a tool from some finite set of tools and may be completed (with possibly different processing times) on any of its set of compatible machines. A pre–schedule obtained by a global scheduling code is followed until a disruption occurs. The match–up heuristic chooses a subset of the machines to reschedule so that the pre–schedule can again be followed in $\bar{l}$ time units. The objective over this horizon includes weighted tardiness and an incentive to keep jobs on their pre–scheduled machines.

The match–up heuristic code has been applied to a variety of situations using actual data from plants. In ten of eleven actual problems, the heuristic was able to obtain significant improvements over the previously used solution of pushing back the pre–schedule (the eleventh problem tied). The average reduction in tardiness was 36%. Due to the proprietary nature of the results, no further details of testing can be reported at this time. Further testing is planned to assess the heuristic's performance in other situations.

In summary, we have presented a method that adapts for a changing production environment by seeking to match up with a pre–schedule. The method becomes optimal as the rescheduling horizon is lengthened and the interval between disruptions increases. This is shown using results from economic turnpike theory. The error involved in using the method is also bounded by the difference between the match–up cost and a lower bound found by local minimization. Comparisons of the error bounds for different match–up problem sizes can be used to determine the value of additional computational effort in specific situations.

APPENDIX

The following technical assumptions are necessary for the theory in McKenzie. While important to some applications, these are technicalities in the scheduling problems discussed in this paper. They are included for completeness.

Finite Transitions: For any given $\alpha < \infty$, there exists $\beta < \infty$ such that $||x_{k-1}|| < \alpha$ implies that $|\phi_k| < \beta$ and $||x_k|| < \beta$ for all feasible π. For weighted tardiness this is clear since no more than one unit of processing can be done in any period and tardiness is bounded by the weights.

Well-Defined States: As noted, costs over the infinite horizon may diverge. Assume that a weakly optimal schedule is at hand. Alter the cost in each period by subtracting the cost of the weakly optimal schedule in that period. Then, the infinite horizon cost of the weakly optimal schedule is zero. A state at some time k is well–defined if the infinite horizon cost beginning at that state and time is finite using this altered cost structure. Any state not well–defined cannot be continued from in finite cost. Given the assumption of sufficient slack to reach the weakly optimal schedule, the set of well–defined states at time k is $\Re^{(n_k)}$.

Uniform Lower Bound on Convexity of Von Neumann Facets: It is sufficient to assume that there is a uniform positive lower bound on the weights used in weighted tardiness or flowtime. (See McKenzie for further details.)

Uniformly Bounded Von Neumann Facets: This is trivial since no more than one unit of processing can be done in any period. (See McKenzie for further details.)

TABLE 1: Example Problem Data

job #	1	2	3	4
proc. time	1	1	1	3
due time	1	3	5	7

TABLE 2: Expected Average Cost Values

Disruption Position	**Down Time**	**Probability**	**Average Cost**
0	1	0.10	0.059
0	2	0.10	0.237
1	1	0.10	0.059
1	2	0.10	0.237
2	1	0.10	0.059
2	2	0.10	0.237
3	1	0.10	0.000
3	2	0.10	0.118
4	1	0.10	0.000
4	2	0.10	0.059
TOTAL			**0.107**

REFERENCES

Baker, K. [1977], "An Experimental Study of the Effectiveness of Rolling Schedules In Production Planning," **Decision Sciences**, Vol. 8, pp. 19–27.

Bean, J., J. Birge, and R. Smith [1984], "Aggregation in Dynamic Programming," Technical Report 84–10, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan 48109.

Blackwell, D. [1962], "Discrete Dynamic Programming," **Annals of Mathematical Statistics**, Vol. 33, pp. 719–726.

Derman, C. [1966], "Denumerable State Markov Decision Processes—Average Cost Criterion," Annals of Mathematical Statistics, Vol. 37, pp. 1545–1554.

Glazebrook, K. [1981], "On Non–Preemptive Strategies in Stochastic Scheduling," **Naval Research Logistics Quarterly**, Vol. 28, pp. 289–300.

Hopp, W., J. Bean, and R. Smith [1984], "Non–Homogeneous Markov Decision Processes," Technical Report 84–24, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan 48109.

McKenzie, L. [1976], "Turnpike Theory," **Econometrica**, Vol. 44, pp. 841–864.

Pinedo, M. [1983], "Stochastic Scheduling with Release Dates and Due Dates," **Operations Research**, Vol. 31, pp. 559–572.

Pinedo, M. and S. Ross [1980], "Scheduling Jobs Subject to Non–Homogeneous Poisson Shocks," **Management Science**, Vol. 26, pp. 1250–1257.

Ramsey, F. [1928],"A Mathematical Theory of Savings," **Economic Journal**, Vol. 38, pp. 543–559.

FIGURE 1: Optimal Paths

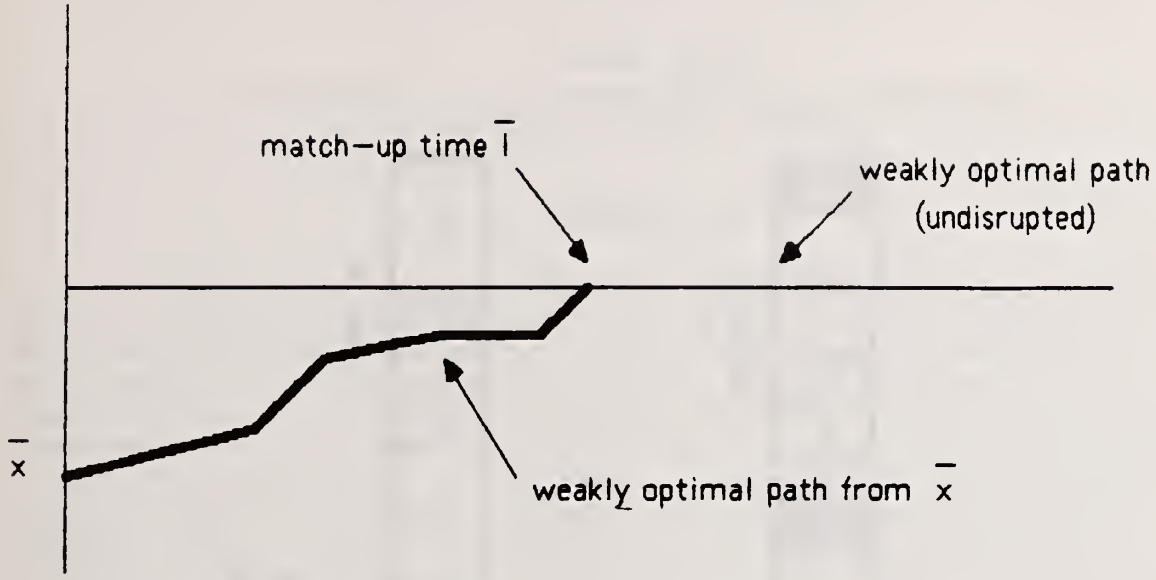
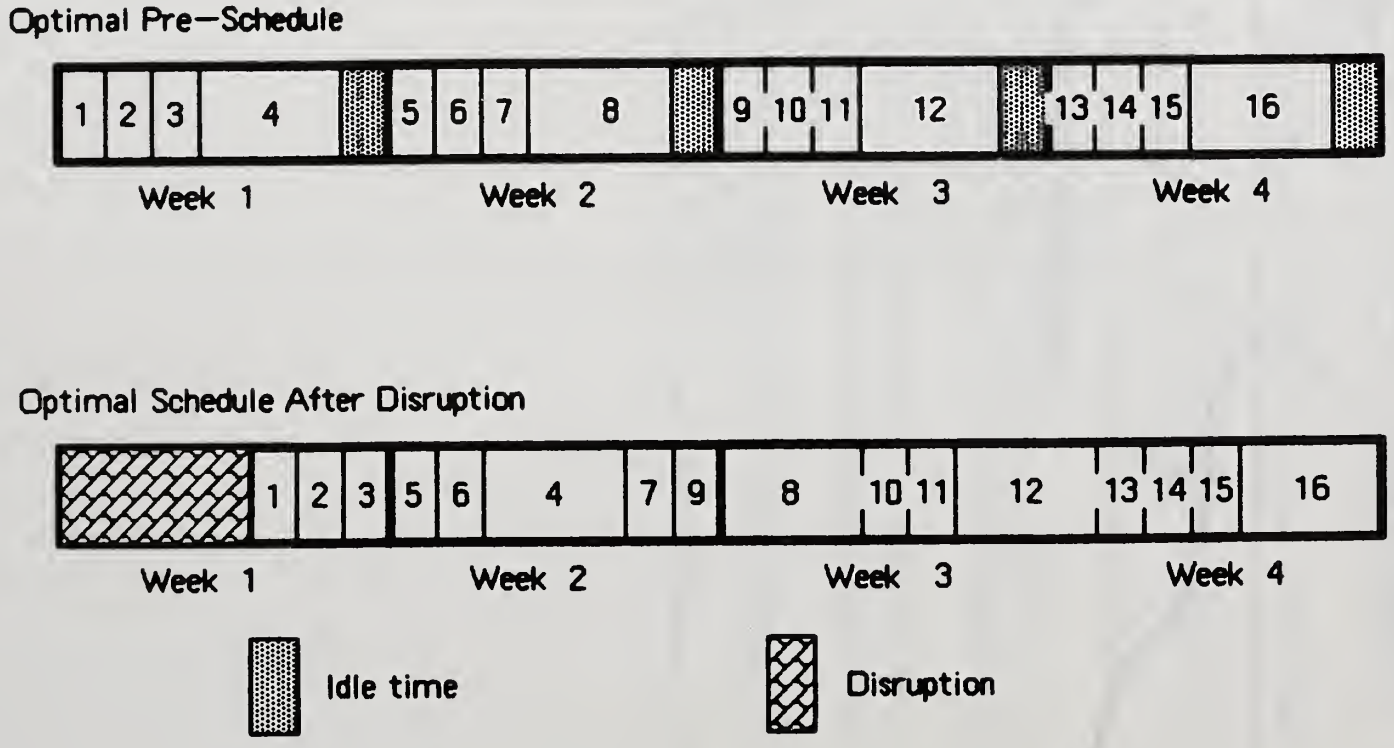


FIGURE 2: Match-Up Example



A MAXIMAL COVERING MODEL FOR LOADING FLEXIBLE MANUFACTURING SYSTEMS

Chen-Hua Chung
Department of Management, University of Kentucky
Lexington, Kentucky, 40506

INTRODUCTION

In recent years, Flexible Manufacturing Systems (FMS) have received much attention from both industrial practitioners and academic researchers. It has been recognized that, in addition to cost and quality, flexibility can also be an important and viable competitive weapon. FMS provides manufacturing firms the capability of achieving cost savings, productivity gains, and quality improvements via the economies of both scale and scope. The aim of an FMS is to achieve the efficiency of (automated) mass production while utilizing the flexibility of a job shop by simultaneously machining several part types (Stecke, 1983). An FMS is an automated (batch) manufacturing system consisting of Numerical Control (NC) machines that perform the operations required to manufacture parts or components. Although the potential benefits of an FMS seem to be enormous, many planning and control problems are yet to be resolved. Among them, the allocation of the operations and required tools of a set of part types among a set of machine (or machine groups) is an important class of problems. This is the so-called "loading problem" in FMS.

In an FMS, each operation would require a set of tools. Each operation can be processed on a certain set of machines, but can be assigned to only one machine. The tools are to be placed in the capacitated tool magazines of the machines. Each tool can be used on different machines with different "coefficients of efficiency".

In loading the FMS, many objectives have been recognized in the literature. For example, Stecke (1983) lists the following six loading objectives:

(1) Balance the assigned machine processing times.

(2) Minimize the number of movements from machine to machine, or equivalently, maximize the number of consecutive operations on each machine.

(3) Balance the workload per machine for a system of groups of pooled machines of equal sizes.

(4) Unbalance the workload per machine for a system of groups of pooled machines of unequal sizes.

(5) Fill the tool magazines as densely as possible.

(6) Maximize the sum of operation priorities.

Which objective is applicable would be problem-dependent. Each may be best under certain circumstances. In some situations, some of these objectives may be conflicting, while in others, several objectives may be equally desirable.

Stecke (1983) formulates several nonlinear mixed integer programs, one for each of the above six objectives. Although various algorithms can be applied to these nonlinear integer programs, the computational burden can be formidable. Furthermore, the accommodation of multiple objectives has not been addressed. One possibility would be the use of some multiple criteria decision making (MCDM) approach. Then, additional issues such as goal preemption may need to be addressed. Since many loading objectives are nonlinear, the MCDM approach may also be computationally formidable.

In this study, we attempt to develop some heuristic algorithms which will efficiently solve an FMS loading problem involving multiple objectives. A heuristic approach will provide satisfactory solutions with considerably less computational effort. Several alternative loading objectives can be accommodated in the same solution process. The difficulties associated with the one-objective-one-model approach can be avoided.

In the following sections, we will first critically assess various loading objectives including the aforementioned six types. This exploration will provide valuable insights into the FMS loading problem. We then present the formulations for a set of selected loading objectives. Heuristic algorithms will then be developed to achieve these objectives.

AN EVALUATION OF LOADING OBJECTIVES

The most adopted loading objective (for both conventional systems and FMS) is to balance the assigned workload on each machine. That is, try to make the total processing times of the operations assigned to each machine as equal as possible.

However, Stecke (1981,1983) points out that the practice of balancing may be too restrictive for most FMSs, since the inherent flexibility can be utilized for better system performance. Specifically, it is found that the expected production is maximized by assigning (i) a balancing workload to each machine, if all group sizes are identical; (ii) a specific, unbalanced workload to each machine, if group sizes are unequal. (These two "rules" were listed as the third and the fourth loading objectives in the previous section).

Existing loading studies usually assume that processing times are known for every operation of all parts. However, since the coefficient of efficiency for each tool differs from machine to machine, the processing time of each part would vary accordingly. In other words, the actual processing time for each operation will be determined by the final loading assignments. To balance the machine loads, one has to take into consideration the tool efficiency when assigning the parts to the machines. Quite often, what is more desirable is to minimize the (total) processing times by maximizing the tool efficiency. Chakravarty and Shtub (1984) address the loading problem with an objective of minimizing the maximum processing time of any of the

machines. Although not a complete substitute, the minimization of the maximum processing time can be a good surrogate for balancing the machine workloads. However, minimizing total processing time and balancing machine workloads can be two conflicting objectives.

The second objective of minimizing the number of movements from machine to machine is important when the transportation time between machines is significant (relative to average operation time). This objective may also be conflicting with the desire of balancing the machine loads. That is, in some cases, it may be more desirable to have a part remain on a machine for several operations rather than have it moved from machine to machine for the sake of balancing loads (Stecke 1983, Stecke and Solberg 1981). Processing time can be reduced by working several operations on the same machine, as long as it is technologically possible. Optimizing this objective may result in large queue times due to overloading some machines.

On the other hand, it may be desirable to pool the operations with common tool requirements. If a small set of tools can perform many different operations, the tool utilization can be increased. However, the required operations of a particular part type may have a diverse tool requirements. Thus, the aforementioned objective of minimizing the number of movements from machine to machine may be in conflict with the desire to take advantage of tool commonality. In a way, the consideration of pooling operations with tool commonality relates to the part-grouping problem. That is, for any two part types, the higher degree of tool requirements commonality, the more "similar" they are and, therefore, the more likely they will be grouped together. In fact, Chakravarty and Shtub (1984) approach the FMS loading problem by first forming workpiece-tool groups and then assigning the groups to machines. The grouping of workpiece-tool is similar to machine-component grouping problem in Group Technology (GT). (See Burbidge (1963), King (1980), King and Nakornchai (1982)).

Stecke (1983) provides a rationale for the fifth loading objective of filling the tool magazines as densely as possible. It is expected that when tool magazines are filled, perhaps several operation assignments may be duplicated to produce alternative part routes, which should increase machine utilization and production, and decrease waiting time (Stecke 1983, p. 281). However, the filling of the tool magazines should be the end result of operation assignments, not the other way around, unless the former is used as a surrogate for the later.

The significance of the sixth objective of maximizing the sum of operation priorities will again depend on situations. Some operations, such as bottleneck operations, may receive higher weights in the loading process. However, it should be noted that the definition (i.e., the occurrence) of bottlenecks will be determined by the processing times of the operations. As previously mentioned, the processing times for each operation will depend on the tool efficiencies which is a function of the loading assignments.

It is true that load balancing is an important loading objective. However, an often overlooked fact in the loading literature is that the tool efficiency plays an important role in loading FMS. First of all, the tool efficiency is a determinant of processing times of the operations. Secondly,

the flexibility of a manufacturing system may be enhanced by reducing processing times via increasing tool efficiency. A strict load balancing policy, on the other hand, may greatly reduce the flexibility. In this study, the tool efficiency will be incorporated into the formulation of various loading objectives.

MODEL FORMULATIONS

FMS loading can be done on a periodic basis (e.g., weekly or daily). As the demand for parts changes over time, the loading decisions (i.e., the assignment of the parts, related operations and the tools to the machines) need to be updated so that the system's efficiency can be improved. Certainly, the loading decisions are closely related to other issues in FMS planning and control. For example, ideally, in making the loading decisions one should take into consideration the characteristics of every operation for each part. However, if the firm has voluminous parts with a great many varieties to be processed, it may consider grouping the parts into part-families. Then, the loading will be carried out in terms of families rather than parts. Although maximum tool efficiency may not be achieved, loading part-families does have its practical meaning from a computational point of view. It may even be a must when the loading decisions are to be made in a real-time fashion. On the other hand, decisions such as sequencing of parts and operations, the routing of operations, etc., are also closely related to the loading problem. However, for practical purpose (e.g., computational consideration), it may be desirable to make these decisions after the loading assignments are determined. Thus, the routing of operations will not be addressed in this study.

In the previous section, we have explored various loading objectives. Most of these objectives are nonlinear in nature. Some of them are conflicting. In this section, we present the formulations of several loading objectives. Although a single model cannot accommodate all types of loading objectives, it does serve well both as a framework for loading decisions and as a foundation for developing the loading heuristics. (For the model formulations of some other loading objectives, see Stecke (1983)).

Quite often, part-family grouping needs to be performed before the loading assignments can be determined. If the number of parts is reasonably small, no grouping will be necessary. In the situation where parts are grouped into families, the loading will be performed on the part-family basis. The model and the heuristics can be easily modified. In the following paragraphs, we will first define the notations used in the model formulations. Then, the part-family grouping problem will be discussed. Several loading objectives will be formulated together with the loading constraints.

Let

$H: \{ h \mid h = 1, \ldots, L_0 \}$, the index set of parts.

$I: \{ i \mid i = 1, \ldots, L_1 \}$, the index set of operations.

$J: \{ j \mid j = 1, \ldots, L_2 \}$, the index set of tools.

K: $\{ k \mid k = 1, \ldots, L_3 \}$, the index set of machines.

G: $\{ g \mid g = 1, \ldots, L_4 \}$, the index set of the "representative parts" for the part-families; each part in H can be a candidate for such representatives.

$a_{ij} = \begin{cases} 1 & \text{if operation i requires tool j,} \\ 0 & \text{otherwise.} \end{cases}$

$b_{hi} = \begin{cases} 1 & \text{if part h requires operation i,} \\ 0 & \text{otherwise.} \end{cases}$

$d_{ik} = \begin{cases} 1 & \text{if operation i can be processed on machine k,} \\ 0 & \text{otherwise.} \end{cases}$

$f_{hk} = b_{hi} * d_{ik} = \begin{cases} 1 & \text{if part h can be processed} \\ 0 & \text{on machine k otherwise.} \end{cases}$

e_{jk} : $0 \leq e_{jk} \leq 1$, the coefficient of efficiency for tool j used on machine k.

t_{ik} : the processing time of operation i on machine k.

p_{ij} : the processing time for operation i while using tool j, assuming the tool is used at its maximum efficiency.

s_{hg} : the "distance" between parts h and g; i.e., the similarity or dissimilarity between the two parts, calculated from a set of attributes of the parts.

S : the maximal (coverage) distance between two parts.

c_k : the capacity of the tool magazine on machine k.

u_g : the upper bound of the size of the part-family represented by the median part g.

m : the number of part-families.

$N_k = \{ i \mid d_{ik} = 1 \}$ the set of operations that can be processed on machine k.

$N_i = \{ k \mid d_{ik} = 1 \}$ the set of machines that can process operation i.

$B_h = \{ i \mid b_{hi} = 1 \}$ the set of operations required by part h.

$B_i = \{ h \mid b_{hi} = 1 \}$ the set of parts that require operation i.

$M_i = \{ j \mid a_{ij} = 1 \}$ The set of tools required to process operation i.

$M_j = \{ i \mid a_{ij} = 1 \}$ the set of operations that require tool j.

$R_h = \{ k \mid f_{hk} = 1 \}$ the set of machines that part h can be routed to.

$P_g = \{ h \mid s_{hg} \leq S \}$ the set of parts that can be represented by part g (if g is the median of a part-family); i.e., the distance between parts h and g is within S.

$X_{ik} = \begin{cases} 1 & \text{if operation i is assigned to machine k,} \\ 0 & \text{otherwise.} \end{cases}$

$Y_{hk} = \begin{cases} 1 & \text{if part h is assigned (routed) to machine k,} \\ 0 & \text{otherwise.} \end{cases}$

$Z_{hg} = \begin{cases} 1 & \text{if part h is represented by part g,} \\ 0 & \text{otherwise.} \end{cases}$

$V_h = \begin{cases} 1 & \text{if part h is covered by (i.e., assigned to) a part-family,} \\ 0 & \text{otherwise.} \end{cases}$

Kusiak (1984) formulates a part-family grouping model based on a clustering model discussed in Arthanari and Dodge (1981). The model (with revised notations) is as follows:

$$\text{Min} \quad \sum_{h=1}^{L_0} \sum_{g=1}^{L_0} s_{hg} \, Z_{hg} \qquad (1)$$

s.t.

$$\sum_{g=1}^{L_0} Z_{hg} = 1 \quad \forall \; h = 1, \ldots, L_0 \qquad (2)$$

$$\sum_{g=1}^{L_0} Z_{gg} = m \qquad (3)$$

$$Z_{hg} < Z_{gg} \quad \forall \; h = 1, \ldots, L_0; \; g = 1, \ldots, L_0 \qquad (4)$$

$$Z_{hg} = 0, 1 \qquad (5)$$

Chung (1986) demonstrates that the clustering problem can be solved by a class of maximal covering location planning model (MCLP). The MCLP formulation for the part-family grouping problem is as follows:

$$\text{Max} \quad \sum_{h=1}^{L} \sum_{g=1}^{L} V_h \qquad (6)$$

s.t.

$$\sum_{h \in P_g} Z_{hg} - V_h = 0 \qquad h \in H \tag{7}$$

$$\sum_{g=1}^{L} Z_{gg} = m \tag{8}$$

$$Z_{gg} - Z_{hg} \geq 0 \qquad h \in P_g \tag{9}$$

If it is desirable to restrict the size of a part-family, a "capacity constraint" can be added:

$$\sum_{h \in P} Z_{hg} \leq u_g Z_{gg} \qquad g = 1, \ldots, L_o \tag{10}$$

The MCLP approach to clustering is to maximize the number of parts covered by (i.e., assigned to) the part-families rather than directly minimize the within-group differences. With the use of the MCLP, the goal of maximizing the within-group homogeneity and between-group heterogeneity is implicitly accomplished by the "maximal (coverage) distance S" mechanism. If the distance between parts h and g is beyond S, then part h will not be assigned to g (if g is the median of a part-family). The smaller the S (pre-)specified for the model, the higher degrees of the within-group homogeneity and the between-group heterogeneity. It is true that different S's will result in different levels of coverage. However, to accomplish total coverage (i.e., to ensure that all parts are assigned to some part-family) we can simply assign each of those parts not covered in the model solution to its nearest cluster median. Thus, the choice of the maximal distance S is rather insignificant in this clustering application.

After the part-family grouping is completed, the loading assignments can be carried out on the part-family basis. However, for simplicity, the discussions in the following paragraphs will assume that the number of parts is reasonably small and no grouping is necessary.

Stecke (1983) formulates several types of objective functions for balancing machine loads (i.e., machine processing times). However, the effects of tool efficiency were not considered. The following objective function recognizes different processing times for an operation being processed on different machines.

$$\text{Min} \sum_{k}^{L_3 - 1} \sum_{k'=k+1}^{L_3} \left| \sum_{i \in n_k} t_{ik} X_{ik} - \sum_{i \in n_{k'}} t_{ik'} X_{ik'} \right| \tag{11}$$

where

$$t_{ik} = \sum_{j \in M_i} p_{ij} \left(\frac{1}{e_{jk}} \right) d_{ik} \tag{12}$$

Since p_{ij} is the processing time for operation i while using tool j with the assumption that tool j is used at its maximum efficiency, the actual processing times may be inflated by $(1/e_{jk})$ if the tool is assigned to a machine where the tool efficiency is less than 100%. The load balancing objective function (1) attempts to minimize the difference in the total processing times between any two machines.

If the minimization of the total processing time of all parts is the major concern, then the following objective function will be appropriate:

$$\text{Min} \quad \sum_{h}^{L_0} \sum_{i \epsilon B_h} t_{ik} X_{ik} \qquad (13)$$

To minimize the number of movements from machine to machine, one necessarily encounters the routing issue. Since the routing problem is not to be addressed in this study, we assume that the times required to move a part from machine to machine are either insignificant or a constant. If the movement times are insignificant, then there is no need to minimize the number of movements from machine to machine. Thus, the load balancing objective function (1) would be sufficient. If the movement times are significant, we would like to assign as many consecutive operations on the same machine as possible. With the assumption of constant movement times, our concern would be simply to minimize the number of movements from machine to machine regardless the sequence of these movements. (In the routing problem with unequal movement times between machine pairs, it is possible that two movements will be more economical than one).

Stecke (1983) uses a somewhat cumbersome method in defining the number of movements from machine to machine. A simple surrogate would be the number of machines a part visits. The smaller the number of machines a part visits, the smaller the number of movements from machine to machine. Thus, the objective function becomes:

$$\text{Min} \quad \sum_{h} \sum_{k \epsilon R_h} Y_{hj} \qquad (14)$$

The constraints in loading models are usually problem-dependent. However, the following two are essential:

$$\sum_{i \,\epsilon\, N_k} \sum_{j \,\epsilon\, M_i} a_{ij} X_{ik} \leq c_k \qquad \forall\, k \qquad (15)$$

$$\sum_{k \,\epsilon\, N_i} X_{ik} \leq 1 \qquad \forall\, i \epsilon B_h, \quad \forall\, h \qquad (16)$$

The first constraint set (5) specifies the capacity limit on the tool magazine of each machine. The second constraint set (6) insures that each operation of a part is assigned to not more than one machine.

As previously mentioned, it is difficult to incorporate multiple and sometimes conflicting objectives in one model. Even with the use of an MCDM approach, issues such as the determination of weights assigned to different

objectives are yet to be addressed. Furthermore, some objective functions are nonlinear. The computational burden may be formidable. In the next section, we develop some heuristic algorithms for the loading problem. The above loading formulations can be viewed as a maximal covering problem. The heuristics presented in the next section are "greedy algorithms" in nature. (Several greedy algorithms have been developed in the MCLP literature, see Chung (1986)).

THE HEURISTIC ALGORITHMS

In this section we present two heuristic algorithms for loading FMS's. The first one focuses on balancing the loads on machines. The second heuristic intends to minimize the movements from machine to machine. Both heuristics take into consideration the maximization of the tool efficiencies and therefore the minimization of processing times.

The Heuristic That Balances the Loads

Step 1: Arbitrarily select a machine k.

Step 2: Compare t_{ik} for each i in N_k .

Step 3: Assign the operation with the lowest t_{ik} to machine k. If there is a tie, or the difference between the processing times of two (or more) candidate operations is minimal (e.g., within $\varepsilon\%$), then the operation that has more required tools already on that machine will receive higher priority.

Step 4: Select another machine and repeat Steps 1 through 3 until all machines are assigned with one operation.

Step 5: Repeat Steps 1 through 4. The machine that has the lowest total processing time of the assigned operations will receive highest priority. The machine with tool magazine capacity exceeded will be excluded from the assignment consideration.

Step 6: Stop when all parts are assigned or when the tool magazine capacity of every machine is exceeded.

The Heuristic That Minimizes the Movements from Machine to Machine

Step 1: Arbitrarily select a part h.

Step 2: Compute t_{ik} for each operation i of part h with respect to each feasible machine k.

Step 3: Define the string of consecutive operations that can be processed on the same machine. Form different combinations of strings such that all required operations for the part can be completed.

Step 4: Compare the total processing time for each combination (i.e., the processing time for the part, $\sum_{i \in B_n} \sum_{k \in N_i} t_{ik}$)

Step 5: Choose the part with the minimum processing time and assign the operations to machines according to the combination of the strings of operations. If any machine has its tool magazine capacity exceeded, choose the combination with the next lowest processing time.

Step 6: Repeat Steps 1 through 5 until all the parts have been assigned or the tool magazine capacity of every machine is exceeded.

CONCLUSIONS

Loading an FMS is an important but complicated task. It usually involves multiple and sometimes conflicting objectives. These objectives are quite often nonlinear in nature. It may be computationally involved in solving the problem optimally. A heuristic approach would be more flexible in incorporating multiple objectives in one solution procedure. It can solve the problem with less computation time although the optimal solution is not guaranteed.

In this study, we explore the nature of the FMS loading problem and develop heuristic algorithms for solving the multiple objective loading problem. Only two sample heuristics are presented. Variations and possible improvements are not discussed here.

Future research should be directed to the further integration of the heuristic algorithms so that more loading objectives can be accommodated simultaneously. Comprehensive performance evaluation of the heuristics under different problem settings would also be needed.

REFERENCES

Ammons, J. C., Lofgren, C. B. and McGinnis (1984) "A Large Scale Work Station Loading Problem", Proc. of The 1st ORSA/TIMS FMS Conf., Ann Arbor, Michigan.

Arthamari, T. S. and Dodge, Y. (1981), Mathematical Programming in Statistics, John Wiley, New York.

Berrada, M. and Stecke, K. E. (1984) "A Branch and Bound Approach to FMS Machine Loading", Proc. of The 1st ORSA/TIMS FMS Conf., Ann Arbor, Michigan.

Burbidge, J.L. (1963) "Production Flow Analysis", Prod. Engnr., 42, 742.

Chakravarty, A.K. and Shtub, A. (1984) "Selecting Parts and Loading Flexible Manufacturing Systems", Proc. of The 1st ORSA/TIMS FMS Conf., Ann Arbor, Michigan.

Chung, C. H. (1986), "Recent Applications of the Maximal Covering Location Planning (MCLP) Model", The J. of Oper. Res. Society, (Forthcoming).

Hankins, S. L. and Rovito, V. P.(1984), "A Comparison of Two Tool Allocation and Distribution Strategies for FMS", Proc. of The 1st ORSA/TIMS FMS Conf., Ann Arbor, Michigan.

King, J. R., (1980) "Machine-Component Grouping in Production Flow Analysis: An Approach Using a Rank Order Clustering Algorithm", Int. J. Prod. Res., 18(2), 213-232.

King, J. R. and Nakornchai, V. (1982) "Machine-Component Group Formation in Group Technology: Review and Extension", Int. J. Prod. Res., 20(2) 117-133.

Kusiak, A., (1984), "The Part Families Problem in Flexible Manufacturing Systems", Proc. of The 1st ORSA/TIMS FMS Conf., Ann Arbor, Michigan.

Stecke, K. E. (1981) Production Planning Problems for Flexible Manufacturing Systems, Ph.D. Dissert. Purdue Univ. West Lafayette, Indiana.

Stecke, K. E. (1983) "Formulation and Solution of Nonlinear Integer Production Planning Problems for Flexible Manufacturing Systems", Mgt. Sci. 29(3), 273-288.

Stecke, K. E. and Solberg, J. J. (1981) "Loading and Control Policies for a Flexible Manufacturing System", Int. J. Prod. Res. 19(5), 481-490.

μ

INTERSTAGE TRANSPORTATION PLANNING IN THE FLOW-SHOP ENVIRONMENT

Michael A. Langston
Department of Computer Science
Washington State University
Pullman, Washington 99164-1210

and

Annette M. Morasch
Hewlett-Packard Company
McMinnville, Oregon 97128

Consider a processing or manufacturing facility in which each piece of work (hereafter referred to as a job) must flow through a linear sequence of k operations. That is, we may think of a job as an ordered list of k tasks, though some tasks may for convenience be vacuous. Similarly, we view the facility as one having k stages. At stage i, $1 \leq i \leq k$, one or more machines exist, each capable of processing the ith task of any job. In general, jobs are independent of each other and tasks, once begun, are not preemptable.

We have just described the well-known flow-shop model, discussed in various contexts (see for example [CD], [CMM] or [Go]). Probably the most frequently cited paper pertaining to flow-shop operation is the solution to the problem of minimizing the overall finish time for a collection of jobs when k=2 and each stage consists of a single machine [Jo]. In that report, as in almost all work published on the flow-shop model, it is assumed that work flows from one stage to the next instantaneously without delay. The only deviations from this assumption appear to be those which concentrate on problems associated with finite buffer length as discussed in papers along the lines of [Ke] or [RR].

Herein we focus on the critical but heretofore ignored issue of transportation delay between adjacent stages, scrutinizing the costs involved at such an interface. We therefore narrow our attention to only two stages of a system, say stages A and B, each having M machines, and assume the existence of a transport which ferrys work from stage A to stage B. This model fits extremely well to a variety of processing situations, from the practical problem of industrial materials handling (see, for example, [NF] for a description of an IBM materials transfer system) to the as yet theoretical issue of multiple robot movement (see, for instance, [LK] for basic research on robot coordination). For this flow-shop model we seek to design and analyze effective transportation planning methods.

Our work is organized as follows. In the first section we consider a static environment, in which a problem instance consists of a collection of jobs whose processing requirements are known in advance, and our objective is to finish them all as soon as possible. We devise and formally analyze transportation planning strategies for various situations, depending on the amount of time needed for transportation relative to the amount required for task processing. In Section 2 we take a different approach, addressing a

dynamic environment in which we face a continuous stream of jobs whose processing requirements are not known in advance but instead are based on a variety of probability distributions. Our objective is then to minimize the delay imposed by the transportation system. We employ a vast collection of computer simulations to achieve empirical estimates of the relative effectiveness of alternate transportation planning schemes. The final section of this paper contains remarks pertinent to ongoing and future investigations of these interstage transportation planning problems.

1. The Static Model

In this section we consider a deterministic version of the flow-shop system, wherein a problem instance consists of a finite set of jobs whose tasks possess known processing times. Our objective is to minimize the latest job completion time. Letting N represent the cardinality of the job set, we denote task requirements for job i, $1 \leq i \leq N$, by $J_i = \langle T_{iA}, T_{iB} \rangle$. We employ a function ℓ to denote the length of each task. For example, if $\ell(T_{1A})=2$ and $\ell(T_{1B})=5$, then the first task of job 1, that for stage A, requires 2 time units to be processed, while the second task of job 1, that for stage B, needs 5. Hence if N=1, we presume that T_{1A} begins at stage A at time 0 and finishes there at time 2. After J_1 is transported to stage B, this movement requiring some time interval t, T_{1B} is processed, beginning at time 2+t and ending at time 7+t. Hence the cost of a solution for this very simple example is 7+t.

A transport (which may be viewed alternately as a type of material handling device or as a primitive robot) works between the two stages. When the processing of a task is completed at a given machine of stage A, the transport is directed to move to that machine, pick up the job, ferry it to whichever machine is desired at stage B, and leave it there. For model simplicity, we assume that machines are equally spaced. We ignore size restrictions on any buffering needed, both at the machine and at the stage level. Similarly, we ignore the time spent loading and unloading each job, as well as the issue of transport acceleration and deceleration, assuming that the time required for the transport to move is directly proportional to the distance it must travel. The figure which follows depicts a flow-shop system with two stages, six machines per stage, and the transport sitting idly between the third pair of machines in each stage. The spacing between each pair of adjacent machines is, say, 10 distance units. We assume a scaling so that one distance unit corresponds to one time unit for the transport to travel.

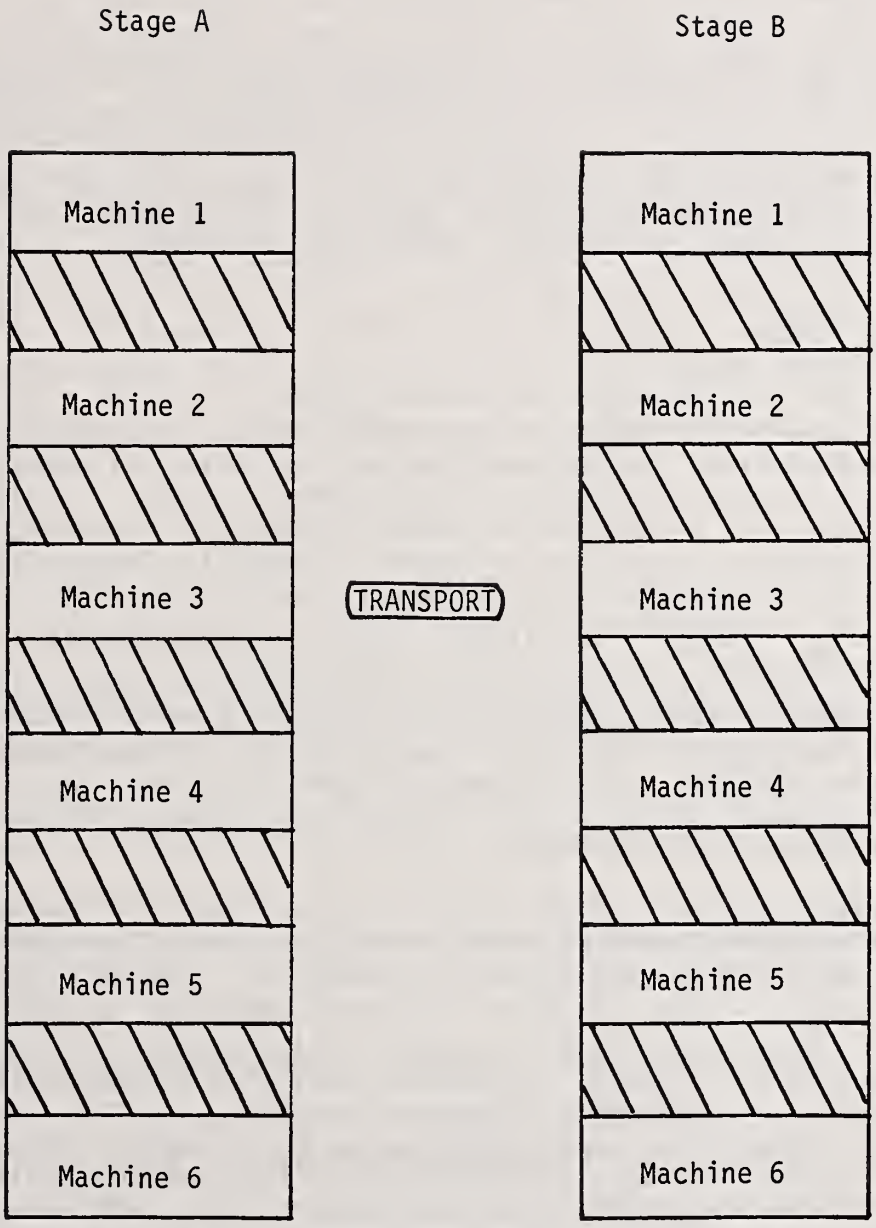


Figure 1. A two-stage system with six machines per stage.

Note that our distance measure is "vertical" rather than "horizontal." That is, using the previous figure as an example, the transport can move a job finishing at machine 3, stage A, to machine 6, stage B, in 30 time units, or to machine 2, stage B, in 10 units, or to machine 3, stage B, in 0 units (essentially a pick-up and put-down operation). Clearly the transport's new position depends on its previous destination.

We consider transportation planning strategies under differing model assumptions. We first address the two extreme cases in which transportation delays are either so small relative to processing times as to be negligible or are so drastically large as to dominate the problem. We then turn our attention to approaches for solving the general problem characterized by transportation times which can neither be ignored nor avoided, but which must be included in planning an efficient transportation scheme.

1.1 Extreme Cases

For the case of negligible transportation costs, the problem takes on a purely scheduling nature and has application in a variety of areas, particularly including computer operating systems software. As mentioned in the introduction, the problem can be solved optimally if each stage contains but a single machine. In such a situation, the familiar Johnson's Rule states that we need merely construct a priority list for the jobs, where J_i precedes J_j if $\min\{\ell(T_{iA}), \ell(T_{jB})\} < \min\{\ell(T_{iB}), \ell(T_{jA})\}$. Since the details of a proof of the rule's optimality can be found elsewhere (see for example [Jo], [CD] or [CMM]), we simply remark at this point that it is a straightforward exercise to show that any optimal list which violates the rule's precedence relation guarantees the existence of an alternate optimal list which obeys the relation. Note that Johnson's Rule is based on a sorting criterion, and hence has a time complexity of $O(N \log N)$.

If we permit multiple machines at each stage, then it turns out that the problem of minimizing the overall completion time, even if transportation costs are negligible, is dramatically more difficult. To see this, suppose that $T_{iB}=0$ for $1 \leq i \leq N$. For this very restricted class of problem instances, we need only decide which job is assigned to which machine at stage A. But what we have done is to reduce our general problem to the special problem generally referred to as the multiprocessor scheduling problem, known to be NP-hard [GJ]. Clearly our general problem can be no easier. Hence no solution procedure is thought to exist which is substantially better than examining all possible schedules and selecting the best. Of course the time required for such a procedure is at least exponential in N, even for the fastest dynamic programming implementations, rendering this approach impractical for problem instances of only moderate size.

Therefore, based on the complexity of this problem, we turn our attention from optimization to near-optimization. We wish to discover fast heuristic or "approximation" algorithms which guarantee solutions whose overall finish times are always close to the minimum. Such an approximation algorithm ALG guarantees, for any problem instance I, a specific "relative performance ratio" R_{ALG} defined as the least real number never exceeded by the ratio $F_{ALG}(I)/F_{OPT}(I)$, where we use $F_{ALG}(I)$ and $F_{OPT}(I)$ to denote the finish times

over I for ALG and an optimization routine respectively. Note that R_{ALG} holds for any problem instance. We will henceforth omit the reference to I when no confusion results.

In this context, we assume that such an algorithm operates by forming a priority list, a specific permutation of the N jobs. This list determines the order in which the jobs are fed into stage A. That is, as a machine at stage A becomes available, the list's next unprocessed stage A task is begun. As work is completed at stage A, jobs are enqueued to stage B on a first-come-first-served basis. Then, when a machine at stage B becomes available, the queue's next unprocessed stage B task is begun.

The curious reader may find it interesting to discover that no algorithm of this type can produce a solution whose overall finish time exceeds three times the minimum, although no smaller ratio will suffice for this algorithmic class. That is, if we let ARB denote an algorithm that arbitrarily constructs a priority list, then $R_{ARB}=3$. Although this ratio has been reported before [BS], and is very similar to a related result from the multiprocessor scheduling literature [Gr], a simple proof of this guarantee at this point will help to illustrate the type of approach used in the remainder of this section.

Theorem 1. $R_{ARB}=3$.

Proof. Our proof that $R_{ARB} \leq 3$ is based on contradiction. Let us assume the existence of a counterexample using a set of jobs J. Thus there is some arbitrary permutation of the elements of J such that $F_{ARB} > 3\ F_{OPT}$. Without loss of generality, let us normalize task lengths so that $F_{OPT}=1$. Therefore $F_{ARB}>3$. Let $X=\langle X_A, X_B\rangle$ denote a job whose processing is completed last in the arbitrary solution. Clearly $\ell(X_A) + \ell(X_B) \leq 1$, else $F_{OPT} > 1$. Also, the processing of X_A must begin on a stage A machine no later than time 1, else the sum of the lengths of all stage A tasks exceeds M, the number of machines at each stage, and again $F_{OPT} > 1$. Hence it must be that X_B is available for processing at stage B no later than time $1 + \ell(X_A) \leq 1 + (1-\ell(X_B)) = 2-\ell(X_B)$. But since the processing of X_B is completed at stage B after time 3, it must not begin there until some time after $3-\ell(X_B)$. Therefore all machines at stage B are busy from time $2-\ell(X_B)$ until some time after $3-\ell(X_B)$, and we conclude that the sum of the lengths of all stage B tasks exceeds M and in any event $F_{OPT} > 1$, which is impossible and contradicts the presumed existence of the counter-example.

To see that $R_{ALG} \geq 3$, consider the following example. Let N=3M-1 and let J contain tasks with these lengths:

$$\ell(T_{1A}) = 0,\ \ell(T_{1B}) = 1$$

$$\ell(T_{iA}) = 0,\ \ell(T_{iB}) = 1/M \text{ for } 1 < i \leq M$$

$$\ell(T_{iA}) = 0,\ \ell(T_{iB}) = 1-1/M \text{ for } M < i < 2M$$

$$\ell(T_{iA}) = 1,\ \ell(T_{iB}) = 0 \text{ for } 2M \leq i < 3M$$

Although $F_{OPT} = 1$ using the original order of J, F_{ARB} can be as large as 3-1/M by reversing the order of J (see Figure 2). In particular, ARB may cause jobs to wait a long time to pass through stage A even if they have little or, in this extreme case, no need for processing at a stage A machine. Letting M grow without bound, we find that no value strictly less than 3 can serve as an upper bound for R_{ARB}. This completes the proof that $R_{ARB}=3$. □

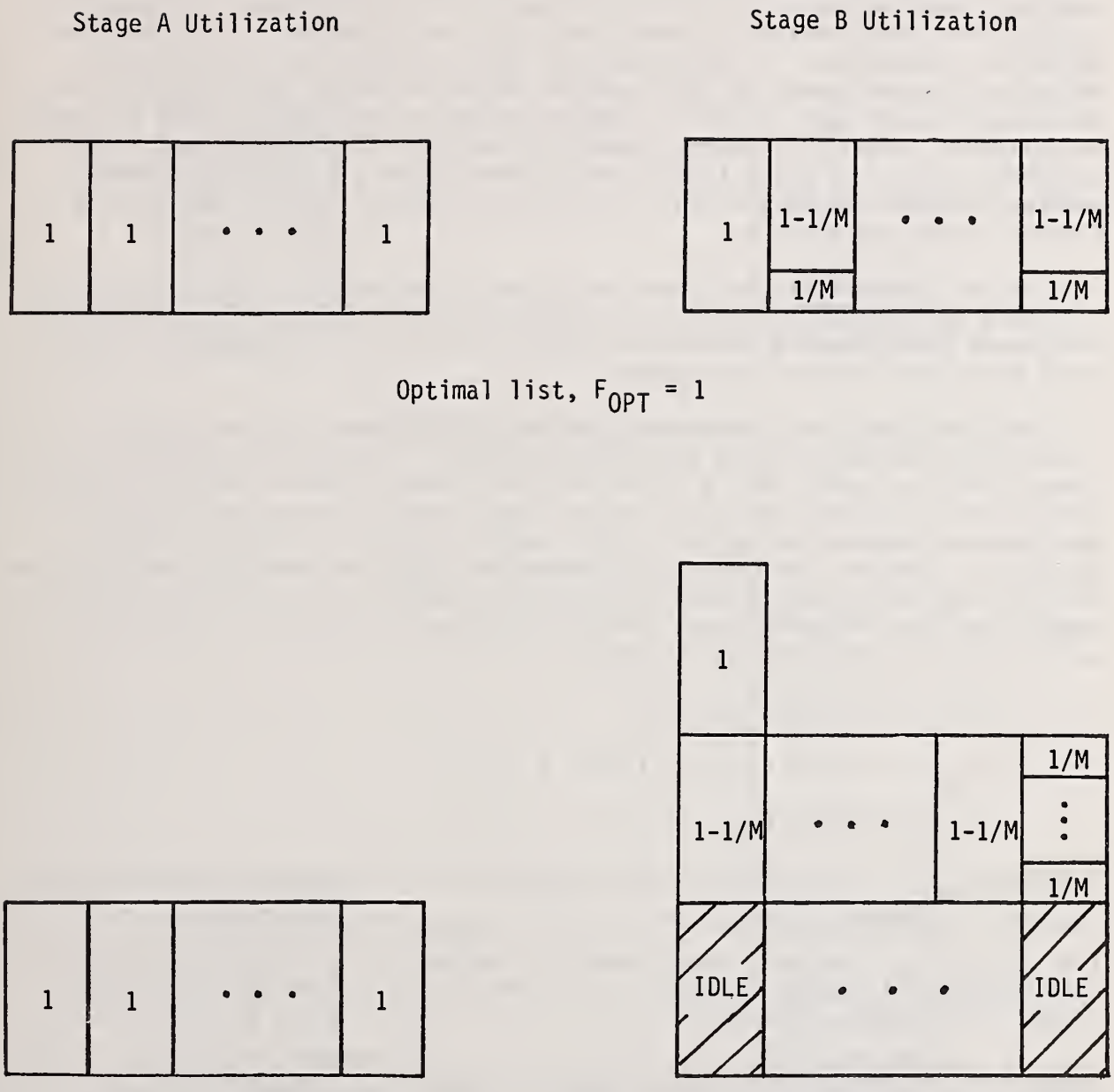


Figure 2. Worst-case instance for arbitrary priority list.

As already mentioned, negligible transportation times and zero processing times at stage B reduce our transportation problem to a parallel scheduling problem. More to the point, non-zero stage B processing times, our real concern, has been previously viewed [BS], [SC] as a problem of both parallel and serial scheduling. In particular, we know from [BS] that the idea of extending Johnson's Rule to this problem, while of course not optimal since the problem is NP-hard, yields a tight performance ratio of 2. That is, such an algorithm, termed MJO which stands for Modified Johnson Ordering, never generates a solution whose finish time exceeds twice the minimum, although problem instances exist for which its solution values approach arbitrarily close to twice the minimum.

We now demonstrate that there is in fact a faster and simpler way to provide a performance ratio of 2. Although Johnson's Rule and hence MJO use both stage A and stage B processing times to construct a priority list, we shall prove that this is unnecessary.

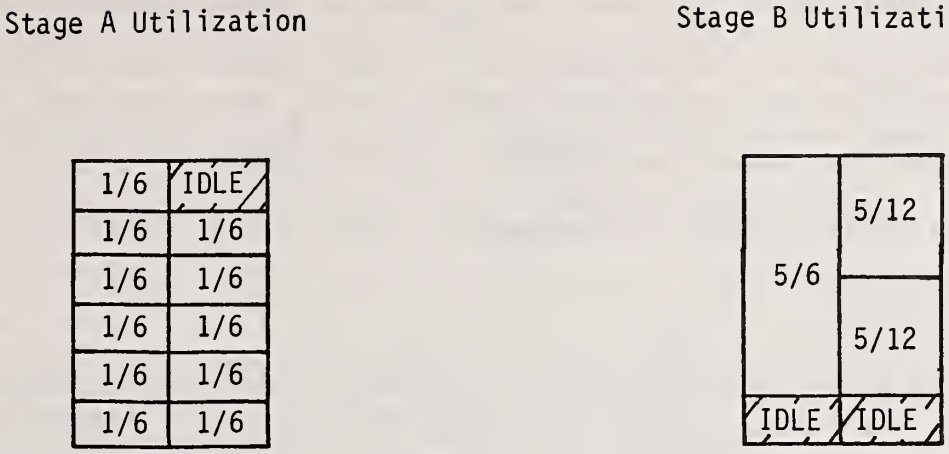
An algorithm which constructs a priority list based only on a non-decreasing sequence of stage A processing times is intuitively appealing. Clearly this is, after all, a motivation for Johnson's Rule: initiate work at stage B as quickly as possible. Let us denote such an algorithm by SORTA. The following example shows that SORTA cannot, however, yield the proper guarantee. (Entire families of such examples exist; we have selected this one for its simplicity. As we have already seen, SORTA can of course yield no completion time exceeding three times the minimum.) Let M=2. Let N=11 and let J contain tasks with these lengths:

$$\ell(t_{1A}) = 1/6,\ \ell(t_{1B}) = 5/6$$

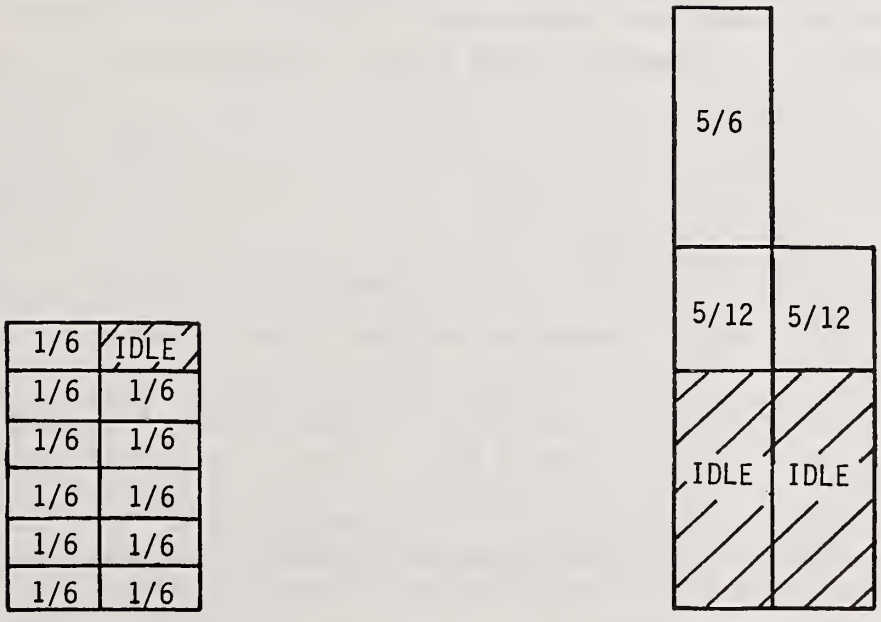
$$\ell(t_{iA}) = 1/6,\ \ell(t_{iB}) = 5/12 \text{ for } i = 2, 3$$

$$\ell(t_{iA}) = 1/6,\ \ell(t_{iB}) = 0 \text{ for } 3 < i \leq 11.$$

Although $F_{OPT} = 1$ using the original ordering of J, the reverse ordering also satisfies the SORTA criteria and therefore F_{SORTA} can be as large as $25/12 > 2$ (see Figure 3). The interested reader may be tempted to consider the use of "tie breakers" to improve SORTA, but one needs only to modify this example slightly to preserve its character (e.g. let $\ell(t_{1A}) = 1/6 + \varepsilon$ and let $\ell(t_{1B}) = 5/6 - \varepsilon$ for some satisfactorily small $\varepsilon > 0$). Furthermore, it is not difficult to see that the type of example illustrated in Figure 3 can be generalized for larger M to demonstrate that SORTA can produce finish times asymptotically as great as 5/2 the optimum.



Optimal list, $F_{OPT} = 1$



SORTA list, $F_{SORTA} = 25/12 > 2$

Figure 3. Simple troublesome instance for SORTA.

Surprisingly, however, we will now prove that we can insure a ratio of 2 by instead preparing our priority list based on a non-increasing sequence of stage B processing times. This algorithm, SORTB, not only provides the appropriate ratio, but the proof of its performance is also much simpler than the corresponding analysis of MJO from [BS]. (Note, however, that in [BS] the number of machines at each stage is independent and is figured into the analysis. We ignore this complication here since it does not affect the asymptotic worst-case behavior of SORTB). SORTB, like MJO, has a time complexity of O(N log N), although SORTB clearly possesses a lower constant of proportionality and is simpler to implement.

Theorem 2. $R_{SORTB} = 2$.

Proof. To prove that $R_{SORTB} \leq 2$, we proceed by contradiction, assuming the existence of a counterexample. We normalize task lengths so that $F_{OPT} = 1$ and hence $F_{SORTB} = 2 + \delta$ for some $\delta>0$. Let $X= \langle X_A, X_B \rangle$ denote a job whose processing is completed last in SORTB's solution. Let t_A represent the time at which processing is initiated for X_A. The set of jobs which precede X in the priority list generated by SORTB keep all M machines at stage A busy without idle from time 0 until time t_A, since otherwise the processing of X_A would begin sooner. Therefore, even in an optimal list, this entire set of jobs can finish processing at stage A no earlier than t_A. Moreover, each such job has a task for processing at stage B whose length is at least as great as $\ell(X_B)$. Thus, since $F_{OPT}=1$, it must be that $t_A + \ell(X_B) \leq 1$. We therefore conclude that in SORTB's solution, X_B is available for processing at stage B no later than time $t_A + \ell(X_A) \leq (1-\ell(X_B)) + (1-\ell(X_B)) = 2 - 2\ell(X_B)$, although the processing of X_B does not commence until time $2 + \delta - \ell(X_B)$. It must then be that all M machines at stage B are continuously busy from time $2 - 2\ell(X_B)$ until time $2 + \delta - \ell(X_B)$, a period of $\ell(X_B) + \delta$ time units.

Consider the first M jobs in SORTB's priority list. Their tasks for processing at stage B make up a collection of the M longest stage B tasks. Let $Y= \langle Y_A, Y_B \rangle$ denote one of these M jobs. Since Y_A begins at time 0 and since $\ell(Y_A) + \ell(Y_B) \leq 1$, the processing of Y_B either begins as soon as Y_B is available at stage B and is thus completed by time 1, or we can associate with Y_B a distinct stage B machine which is busy processing stage B task(s), which are not from this collection of the M longest tasks, from the time Y_B becomes available until it's processing begins. Hence by time 1, the total processing performed at stage B is at least as great as the sum of the lengths of the M longest stage B tasks.

Since strictly fewer then M tasks can finish processing after time 1 at stage A, there is at least one stage B machine that is continuously busy from time 1 until at least time $2 + \delta - \ell(X_B)$, processing only stage B tasks available no later than time 1. Let $Z= \langle Z_A, Z_B \rangle$ denote the last job whose stage B task begins processing on such a machine before time $2 + \delta - \ell(X_B)$.

We now claim that $\ell(X_B) \leq 1/2$. Suppose otherwise. Then X_B must be one of the M longest stage B tasks, else there are at least M+1 stage B tasks each of whose length exceeds 1/2 and $F_{OPT} > 1$. Hence X_B is available at time $\ell(X_A)$, but is not processed until time $2 + \delta - \ell(X_B) \geq 2 + \delta - (1-\ell(X_A)) = 1 + \delta + \ell(X_A)$. But this means that all stage B machines are continuously busy for a period of at least $1 + \delta$, which is impossible.

We next claim that Z_B cannot be longer that the Mth longest stage B task. Suppose otherwise. Hence Z_B is available at time $\ell(Z_A)$ but not processed until at least time $2 + \delta - \ell(X_B) - \ell(Z_B) \geq 2 + \delta - \ell(X_B) - (1-\ell(Z_A)) = 1 + \delta - \ell(X_B) + \ell(Z_A)$. We therefore find two periods during which all stage B machines are continuously busy: one from time $\ell(Z_A)$ to time $1 + \delta - \ell(X_B) + \ell(Z_A)$ and a second from time $2 - 2\ell(X_B)$ to time $2 + \delta - \ell(X_B)$. The sum of these periods is $1 + 2\delta$. Thus we are done unless these periods overlap by at least 2δ time units, implying $2 - 2\ell(X_B) + 2\delta \leq 1 + \delta - \ell(X_B) + \ell(Z_A)$, from which we derive $1 < 1 + \delta \leq \ell(X_B) + \ell(Z_A) \leq 1/2 + \ell(Z_A) < \ell(Z_B) + \ell(Z_A)$, which is impossible.

We conclude the upper bound section of this proof by noting that, since Z_B was available at time 1 but not processed until at least time $2 + \delta - \ell(X_B) - \ell(Z_B)$, all M stage B machines were continuously busy for a period of $1 + \delta - \ell(X_B) - \ell(Z_B) > 0$. If this busy period overlaps with the one beginning at time $2 - 2\ell(X_B)$, then the total processing performed at stage B is at least $M\ell$(the Mth longest stage B task) $+ M(2 - 2\ell(X_B) - 1) + M(\ell(X_B) + \delta) + \ell(X_B) > M\ell(X_B) + M(1 - 2\ell(X_B)) + M(\ell(X_B) + \delta) > M$, which is impossible. On the other hand, if these two busy periods do not overlap, then the total processing performed at stage B is at least $M\ell$(the Mth longest stage B task) $+ M(1 + \delta - \ell(X_B) - \ell(Z_B)) + M(\ell(X_B) + \delta) + \ell(X_B) \geq M\ell(Z_B) + M(1 + \delta - \ell(X_B) - \ell(Z_B)) + M(\ell(X_B) + \delta) > M$, which is once again impossible.

To see that $R_{SORTB} \geq 2$, consider the following example, which we have merely adapted from the multiprocessor "list scheduling" result of [Gr]. Let N=2M-1 and let J contain tasks with these lengths:

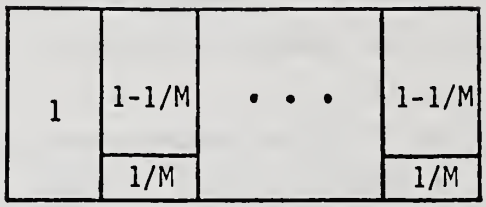
$$\ell(T_{1A}) = 1,\ \ell(T_{1B}) = 0$$

$$\ell(T_{iA}) = 1/M,\ \ell(T_{iB}) = 0 \text{ for } 1 < i \leq M$$

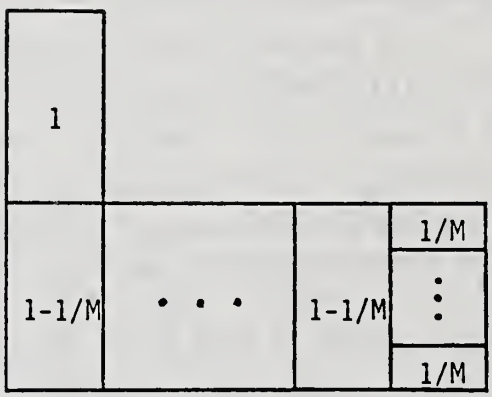
$$\ell(T_{iA}) = 1-1/M,\ \ell(T_{iB}) = 0 \text{ for } M < i \leq 2M-1$$

Although the given job list allows that $F_{OPT} = 1$, the reverse ordering also obeys the criteria of SORTB and thus F_{SORTB} can be as large as 2-1/M (see Figure 4). Letting M grow without bound, we find that no value strictly less than 2 can serve as an upper bound for R_{SORTB}. (Incidentally, this example also illustrates that $R_{MJO} \geq 2$.) This completes the proof that $R_{SORTB}=2$. □

Stage A Utilization
(Stage B Not Utilized)



Optimal list, $F_{OPT} = 1$



SORTB list, $F_{SORTB} = 2-1/M$

Figure 4. Worst-case instance for SORTB.

For the case of truly dominant transportation costs, an optimization strategy is conspicuously apparent. Since distances are large enough to rule out any use of the transport, except of course for its pick-up and place-down activities in one location, the user's best choice is also the simplest: use only one machine, say the first, at each stage and employ Johnson's Rule to produce a priority list.

1.2 The General Case

Suppose now that we allow transportation times to take on any range of values. Clearly our multiple machine problem is NP-hard, having as a special case the set of extreme instances in which transportation delays are negligible. Our ability to analyze the behavior of a fast approximation algorithm is, for this environment, complicated by the fact that we do not even know how many machines are employed at each stage in an optimal solution. If transportation costs are relatively minor, then it may be that most or all are used; if costs are very significant, then only a few or even just one may suffice. As a consequence, the interested reader should have no difficulty in seeing that if we lock our attention on a fixed set of machines, then one can contrive problem instances which show that any heuristic using exactly that set cannot provide any constant performance ratio.

The upshot of these considerations is then that, in an effort to guarantee a constant relative performance bound, our approach will be to apply a "compound" algorithmic strategy [La], in which we produce a series of solutions, one employing a single machine at each stage, another using two, then three and so on. If we have a total of M machines available at each stage, then we choose the best solution from the set of M solutions generated. This is not only a fast and appealing approach, but it greatly simplifies our task of analyzing its effectiveness as well, since we must have generated at least one solution which uses an optimal number of machines and which gives an upper bound to our final selected solution value, regardless of its machine utilization. Hence we need only compare our solution which uses, say, h machines at each stage to an optimal solution using h machines per stage. Now to avoid egregious worst-case behavior, we must avoid making task assignments based solely on stage A or solely on stage B processing times. That is, we seek to ensure that we need traverse the h-1 intermachine distances only once. In order to do this, we direct that a job is assigned the same machine at each stage. Note that without this restriction, it is easy to construct families of examples in which an unbounded number of transport movements are needed, precluding the establishment of any constant performance ratio.

Based on these observations, therefore, our compound approximation algorithm COMP can be described as follows. For each value of h, $1 \leq h \leq M$, we list schedule T to h machine "pairs." That is, J_i is viewed as having a single value namely $T_{iA} + T_{iB}$, and we make job assignments as if there were but one virtual stage and h virtual machines. As a virtual machine becomes available, we assign the next unprocessed job to it. Having thus partitioned J into h subsets, each subset is assigned a machine pair based on a non-decreasing sequence of stage A processing requirements. That is, a subset with the smallest stage A sum is assigned to the lowest indexed machine

pair. The subset with the next smallest stage A sum is then assigned to the next lowest indexed machine pair. This procedure continues until each of the h subsets has been assigned. We now direct the transport to begin at the lowest indexed machine pair. When it has finished moving work to stage B there, it moves then to the next lowest indexed machine pair and so on. This strategy, although simple and of a brute force nature, is quite fast, having time complexity O(MN log M). More importantly, it yields the desired result, a constant performance ratio, as we now show.

Theorem 3. $R_{COMP} \leq 4$.

Proof. Consider any problem instance. Normalize task lengths so that $F_{OPT} = 1$. Let Δ denote the intermachine distance. In an optimal permutation of J, the sum of the lengths of the tasks assigned to any particular machine pair cannot exceed 2, even with maximum concurrency. Hence the sum of the lengths of all tasks is at most 2h, where h denotes the maximum number of machines used at either stage in an optimal schedule. Since the lengths of the tasks of any job sum to a value less than or equal to 1, our list scheduling rule insures that, in any COMP solution, the difference between the maximum and minimum machine pair sums is at most 1. In particular, for COMP's solution on h machine pairs, the minimum cannot exceed 2, else the overall task sum exceeds 2h. Thus for this solution the maximum machine pair sum is at most 3, even with no concurrency.

Now let us focus on one of COMP's h subsets whose processing is completed last. Let P denote such a subset. Let P_A and P_B denote the sum of the lengths of P's stage A and stage B tasks, respectively. The longest possible delay, D, that P_B can experience due soley to the movement of the transport is $(h-1)\Delta$. Any additional delay incurred because the transport must wait for the completion of the processing of stage A tasks at lower indexed machine(s) is absorbed during the processing of P_A since every subset's stage A tasks commence processing at time 0. Therefore, the finish time for P is at most $D + P_A + P_B \leq (h-1)\Delta + 3$. But an optimization rule must also direct the transport to visit the h machine pairs, and $(h-1)\Delta \leq 1$. Therefore $R_{COMP} \leq 4$. □

In establishing a lower bound on the performance ratio of R_{COMP}, one finds that for COMP to generate a remarkably poor solution relative to the optimum, an optimization routine needs to be able to exploit potential concurrency both in transport movement and in stage A and stage B processing. But these goals seem to be diametrically opposed to one another, and we can only say that $R_{COMP} \geq 2$. As a simple example of this, let M=2, N=3, and task lengths be:

$\ell(T_{1A}) = 0$, $\ell(T_{1B}) = 1$

$\ell(T_{2A}) = 1$, $\ell(T_{2B}) = 0$

$\ell(T_{3A}) = 1-\Delta$, $\ell(T_{3B}) = 0$, for some arbitrarily small intermachine distance Δ.

In an optimal partition, J_1 and J_2 can share a machine pair. The transport can

move J_1 at time 0, move J_3 at time $1-\Delta$, and still be able to return to its starting position so as to move J_2 at time 1. Hence $F_{OPT} = 1$. But, even using both machines at each stage, COMP may partition J_2 and J_3 together, in which case $F_{COMP} = 2 - \Delta$. Therefore R_{COMP} must exceed any value strictly less than 2. We conjecture at this point that the exact value of R_{COMP}, lying the the range [2,4], is in fact closer to the lower end of this range, our upper bound leaving considerable slack.

We close this section with a few remarks about machine spacings. Recall that we originally specified that machines be equally spaced for model simplicity. Note that this assumption is of no consequence for the extreme cases of negligible and dominant transportation costs. More importantly, we can handle arbitrary machine spacing in the general case. One need merely specify that for each value of h, $1 \leq h \leq M$, the h closest machines be employed by COMP.

2. The Dynamic Model

Instead of focusing solely on the static problem environment we have just examined, one might naturally ask questions about a dynamic model, in which a continuous job stream flows through the flow-shop. Precise processing demands are not known in advance, though some knowledge of a probability distribution over the task lengths is likely, based either on a priori information about the type of jobs in the input stream or on the observations of past behavior. In the sequel, we will consider task lengths as a set of independent, identically distributed random variables having three distinct probability distributions. We use the uniform distribution to represent an environment in which little is known to help predict the length of a given task. We employ the normal distribution to model a situation in which most task lengths tend to occur in some middling range. Finally, we use the exponential distribution to represent the situation, common at least in computer software systems, in which the majority of tasks have short lengths, and in which large and even very large lengths do occur, although with relative infrequency.

For consistency of our flow-shop model, we leave our transport system unchanged from that described in the previous section. The dynamic nature of this environment, however, suggests that our attention should not be directed to minimizing the overall finish time, but rather to minimizing the delay imposed by the transportation planning system. That is, a job can incur a delay from the time it finishes processing at stage A until the time it begins processing at stage B, even though one or more stage B machines may be available. It is the average transport-induced delay we seek to reduce, thus increasing system thruput, decreasing the average time a job is in the flow-shop, and, additionally, reducing the likelihood of troublesome queueing delays should multiple jobs simultaneously wait on the transport. Roughly speaking then, in this environment, the transport becomes our single critical resource over whose movement we must exercise control.

We observe that the investigation of transportation planning strategies is only meaningful in "draw" systems (i.e., systems in which, on the average, the length of a stage A task exceeds that of a stage B task). Otherwise, we face the expectation of an "unstable" system, in which a queue of unbounded

length forms in front of stage B in the steady state. See, for example, [Tr] for a detailed discussion of the "traffic intensity" parameter, which in this environment denotes the ratio of the mean length of a stage B task to the mean length of a stage A task. Hence, if this ratio is not strictly less than unity, a long queue between stages precludes the effectiveness of any transportation planning strategy. Recall that it is precisely the planning strategies we seek to study, not queueing disciplines which are already a subject of extensive investigation.

In this model we are interested in the long-term, steady state effectiveness of transportation planning schemes. Therefore our desire is to investigate average-case behavior rather than to establish worst-case ratios. Due to the complexity of this model and the many variables affecting average-case performance, we turn from formal analysis to the use of system simulation in an effort to generate a vast collection of empirical observations. To implement our model, we have employed the GPSS/H simulation software [HC] running on an Amdahl 470-V8 mainframe under the VM/CMS operating system. This package and run-time environment was satisfactory for all but the planning strategies themselves, which are to be described later. These we encoded in the FORTRAN programming language and then we linked to them from GPSS/H. Additional implementation details are omitted here for the sake of brevity, but a full version of a report describing our simulation system may be obtained from the authors.

We now present the major transportation planning algorithms on which we have gathered considerable computational experience. Each seeks to move jobs from stage A to stage B efficiently. In order to prevent any single job from being delayed a disproportionately long period of time, all algorithms service jobs strictly on a first-come-first-served basis. Once the transport picks up a job it will move it to the stage B machine on which the job's stage B task can begin processing the soonest, ties to be broken in favor of the machine with the lowest index. Notice that the earliest start time on a machine at stage B depends both on the tasks, if any, already at that machine and on the distance the transport must move in order to arrive at that machine.

Strategy 1. Our first strategy is both simple and unassuming. Since we do not know to which machine at stage A the transport will be ordered next, the transport merely remains stationary when no jobs are waiting to be moved. That is, the transport, after ferrying a job to stage B, stays where it is until it is directed to move to another machine. Observe that this strategy directs no unnecessary transport movement.

Strategy 2. The motivation for our next algorithm relies on an attempt to avoid the worst-case scenario in which a stage A task terminates and the transport must move the entire (vertical) length of the flow-shop to load the job. We specify that the transport move to the middle of the shop when no movement requests are pending.

Strategy 3. Our third approach plays a "guessing game." Although we do not know which machine at stage A will next finish processing a task, we do know when each started its current task. We direct the transport to move, if it has no jobs awaiting movement, to the position of the machine at stage A

which earliest began to process its current task, ties to be broken in favor of the machine with the lowest index.

In order to gauge the effectiveness of these various and sometimes opposing strategies, we also implement a look-ahead rule, which in some sense "cheats" by directing the transport to move to the machine at stage A which will next complete the processing of a task. Of course we have assumed throughout that this information is not available in our dynamic and probabilistic environment; we provide it to our look-ahead rule solely as a means of identifying an "informed" scheme to which we can compare strategies one through three.

For our first series of empirical results, we adopt a uniform probability distribution from which to draw task lengths. We arbitrarily set an intermachine distance of 10 units (that is, the transport requires 10 time units to move from one machine to an adjacent one, twenty to the next one and so on). For this scenario, as well as for those which follow, we have made literally hundreds of simulation experiments. Table I lists a representative sample of performance figures we have obtained for each of the three algorithms under differing traffic intensities, task length means and numbers of machines per stage. Each simulation consisted of 100,000 jobs, enough to eliminate initial perturbations, insuring that we observed true steady state behavior.

For each of these simulations, the total delay charged to a transportation planning strategy is the cumulative length of time, summed over every stage B machine, during which both the machine is idle and one or more jobs, having completed processing at stage A, is scheduled to be moved to that particular stage B machine. The numbers shown under the "excess delay" heading have been computed as follows. We have taken the total delay incurred using one of our strategies, divided it by the total delay of the look-ahead rule on the same job set, multiplied this quantity by 100, rounded to the nearest integer and then subtracted 100, regarding the final value as a percentage. Thus a figure of 25 means that an algorithm causes 125% the delay of the look-ahead rule, that is, 25% in excess of the informed strategy.

Table I. Uniform Distribution

Machines per Stage	Average Task Length A	B	Traffic Intensity	Excess Delay by Strategy 1	2	3	Strategy Rankings
2	50	5	1/10	568	411	321	3,2,1
2	20	5	1/4	131	88	73	3,2,1
2	80	40	1/2	380	275	185	3,2,1
2	40	30	3/4	178	132	95	3,2,1
2	80	70	7/8	204	144	126	3,2,1
4	150	15	1/10	292	207	208	2,3,1
4	80	20	1/4	79	57	59	2,3,1
4	200	100	1/2	219	159	150	3,2,1
4	120	90	3/4	72	55	54	3,2,1
4	200	175	7/8	103	82	78	3,2,1
10	500	50	1/10	81	59	69	2,3,1
10	600	150	1/4	114	84	95	2,3,1
10	800	400	1/2	150	111	123	2,3,1
10	900	675	3/4	115	86	94	2,3,1
10	720	630	7/8	52	39	44	2,3,1
40	9000	900	1/10	120	93	110	2,3,1
40	8000	2000	1/4	69	52	64	2,3,1
40	7000	3500	1/2	42	31	39	2,3,1
40	10000	7500	3/4	144	114	130	2,3,1
40	8000	7000	7/8	42	31	40	2,3,1

Table II illustrates the behavior representative of what we have observed for the transportation planning algorithms under the adoption of a normal probability distribution for task lengths. As before, we set the intermachine distance at 10, used 100,000 jobs to address the steady-state behavior of each strategy, and computed excess delay as the percentage exceeding the optimum.

Table II. Normal Distribution

achines er Stage	Average Task Length A	B	Traffic Intensity	Excess Delay by Strategy 1	2	3	Strategy Rankings
2	50	5	1/10	834	489	194	3,2,1
2	20	5	1/4	86	53	42	3,2,1
2	80	40	1/2	1336	769	271	3,2,1
2	40	30	3/4	362	226	73	3,2,1
2	80	70	7/8	242	171	78	3,2,1
4	150	15	1/10	402	258	182	3,2,1
4	80	20	1/4	137	92	82	3,2,1
4	200	100	1/2	530	337	220	3,2,1
4	120	90	3/4	135	100	69	3,2,1
4	200	175	7/8	112	86	54	3,2,1
10	500	50	1/10	107	77	82	2,3,1
10	600	150	1/4	149	107	110	2,3,1
10	800	400	1/2	232	164	158	3,2,1
10	900	675	3/4	171	126	117	3,2,1
10	720	630	7/8	65	50	48	3,2,1
40	9000	900	1/10	126	96	112	2,3,1
40	8000	2000	1/4	79	58	69	2,3,1
40	7000	3500	1/2	54	40	47	2,3,1
40	10000	7500	3/4	162	127	144	2,3,1
40	8000	7000	7/8	45	34	39	2,3,1

For our last set of empirical results, summarized in Table III, we employed the exponential distribution in order to generate task lengths. Again, an intermachine distance of 10 units was used, 100,000 jobs were generated per simulation, and excess delay was computed a percentage above the look-ahead rule.

Table III. Exponential Distribution

Machines per Stage	Average Task Length A	Average Task Length B	Traffic Intensity	Excess Delay by Strategy 1	Excess Delay by Strategy 2	Excess Delay by Strategy 3	Strategy Rankings
2	50	5	1/10	400	359	363	2,3,1
2	20	5	1/4	85	68	70	2,3,1
2	80	40	1/2	188	173	168	3,2,1
2	40	30	3/4	89	82	80	3,2,1
2	80	70	7/8	114	115	125	1,2,3
4	150	15	1/10	223	172	213	2,3,1
4	80	20	1/4	48	37	45	2,3,1
4	200	100	1/2	136	111	132	2,3,1
4	120	90	3/4	38	31	37	2,3,1
4	200	175	7/8	80	68	82	2,1,3
10	500	50	1/10	60	45	58	2,3,1
10	600	150	1/4	89	67	86	2,3,1
10	800	400	1/2	115	88	111	2,3,1
10	900	675	3/4	79	60	77	2,3,1
10	720	630	7/8	29	20	27	2,3,1
40	9000	900	1/10	112	87	107	2,3,1
40	8000	2000	1/4	61	46	59	2,3,1
40	7000	3500	1/2	34	26	32	2,3,1
40	10000	7500	3/4	125	99	119	2,3,1
40	8000	7000	7/8	20	15	20	2,3,1

As one might expect, we find from Tables I, II and III that strategy 1 almost is never the best of the three tested algorithms. It is frequently, however, at least reasonably competitive with the others. This suggests that if extraneous transport movement incurs a non-zero cost, then this strategy may well be worth a second look. (Recall that strategy 1 requires no unnecessary transport movement.) These simulation results bear out our expectation that as the traffic intensity increases, transportation planning strategies can become nearly indistinguishable from one another as a queue of jobs awaiting the transport forms between flow-shop stages. Notice also that, in general, large stage A task length means tend to inflate the absolute excess delay values for each tested algorithm, since the informed strategy has more lead time, on the average, for ideal transport positioning.

We observe from Table III that, for the exponential distribution, strategy 2 is almost always more effective than strategies 1 and 3. This is pretty much as one might predict, since the expected completion time for each task, as generated by the exponential distribution, is independent of the amount of time it has already spent in processing (see, for example, [CD] for a discussion of this memoryless or Markov property). Therefore, at any time in the steady state, all stage A machines have an equal likelihood of finishing next. Directing the idle transport to proceed toward the center of the flow-shop is a sound scheme for this environment.

At issue, then, remains the determination of when, if ever, it is better to guess, in the sense of strategy 3, than it is to try to play it safe, as done by strategy 2. For both the uniform and the normal distributions, the stage A machine most likely to finish next is the one that has been processing its current task for the longest period of time. (Note that the bell-shaped graph of the normal distribution suggests that it might be even better suited for exploiting strategy 3 than is the uniform distribution.) Despite this observation, however, it is not at all clear that employing strategy 3 is a relatively good idea. In particular, the costs incurred whenever the transport makes an error and proceeds in the wrong direction could well outweigh the savings accrued from correct guesses and from guesses that at least move the transport closer to the proper machine. Our experimental results, as seen in Tables I and II, indicate that in fact for both distributions strategy 3 is likely to be superior to strategy 2 when there are not too many machines in the flow-shop. But as the number of machines grows, the cost of an error grows as well, until a "break-even" point is reached at which strategy 2 gains parity. As one then continues to add machines to the flow-shop, strategy 2 becomes the better alternative. This break-even point appears to occur earlier for the uniform distribution, at around four machines, then it does for the normal distribution, at around ten machines.

3. Conclusions and Directions for Future Research

We have designed algorithms and formally derived their worst-case performance guarantees for the problem of planning and coordinating movement within our flow-shop system in the deterministic, static environment. Though problem complexity often prevents one from devising practical optimization strategies, we have shown how to guarantee near-optimal movement costs for

various situations, depending on the time required for transport movement relative to that for task processing.

We have also investigated transportation planning strategies for the probabilistic, dynamic environment. Computational experience seems to suggest that in the absence of other information, the idle transport can be well-utilized, in the case of the exponential distribution, by blindly try to "cut its losses," heading for the center of the flow-shop. The same result holds for the cases of the uniform and normal task length probability distributions, although one can do even better by trying to "out guess" the system, heading for the machine at stage A that's been processing its current job the longest, if the number of machines at each stage is not too large.

In our approach to the exploration of this model's utility, we have attempted to address several crucial issues. However, many open problems remain unanswered, a few of which we mention at this time. Note that our system bottleneck has, in general, been the assumption of a single unit-capacity transport. Although this has kept the model attractively simple (and inexpensive), one might ask about the effectiveness of movement strategies given multiple transports, or even multiple-capacity transports. Similar problems have, for instance, been considered [CCF] for two-server computer storage systems, although without the notion of physically transporting items between devices or stages.

Also, we have typically specified a first-come-first-served order for job movement. A more complex model might incorporate a priority structure, to be used when multiple jobs await transportation, which could lower the average transportation delay, although a small proportion of the jobs may suffer longer waiting times. Similarly, the model can be modified slightly by permitting different numbers of machines at each stage, or by dropping the assumption that machines are equally spaced within a stage. Naturally, the model could remain unaltered while better static and dynamic strategies are considered (e.g., the worst-case bound of 4 for algorithm COMP in Section 1 may leave considerable room for improvement; it would also be interesting to learn something about the expected behavior of SORTB relative to that of MJO). Finally, we remark that systems with three, four and even more stages, not just extensions to the two-stage model, have yet to be considered.

Acknowledgment
This research is supported in part by the National Science Foundation under grant ECS-8403859.

REFERENCES

[BS] Buten, R. E. and Shen, V. Y. "A scheduling model for computer systems with two classes of processors," Proc. 1973 Sagamore Computer Conf. on Parallel Processing, Springer-Verlag, NY, 130-138.

[CCF] Calderbank, A. R., Coffman, E. G., Jr. and Flatto, L. "Sequencing problems in two-server systems," Bell Labs Tech. Memo, Murray Hill, NJ, 1984.

[CD] Coffman, E. G., Jr. and Denning, P. J. Operating Systems Theory, Prentice-Hall, NJ, 1973.

[CMM] Conway, R. W., Maxwell, W. L. and Miller, L. W. Theory of Scheduling, Addison-Wesley, MA, 1967.

[GJ] Garey, M. R. and Johnson, D. S. Computers and Intractability, Freeman, CA, 1979.

[Go] Gonzalez, M. J., Jr. "Deterministic processor scheduling," ACM Computing Surveys 9 (1977), 173-204.

[Gr] Graham, R. L. "Bounds on multiprocessing timing anomalies," SIAM J. Appl. Math. 17 (1969), 416-429.

[HC] Henriksen, J. O. and Crain, R. C. GPSS/H User's Manual, Wolverine Software Corp., Annandale, VA, 1983.

[Jo] Johnson, S. M. "Optimal two- and three-stage production schedules with setup times included," Nav. Res. Log. Quart. 1, (1954).

[Ke] Kelley, F. P. "Segregating the input to a series of buffers," Math of Op. Res. 10 (1985), 33-43.

[La] Langston, M. A. "Iterative, hybrid and compound strategies for discrete near-optimization," to appear.

[LK] Langston, M. A. and Kim, C. E. "Movement coordination for single-track robot systems," Proc. 1985 IEEE Conf. on Robotics and Automation, St. Louis, MO, 523-529.

[NF] Noller, D. and Ferriter, K. "Animated Micro-Computer Simulation with PC MODEL," Proc. Comp. & Ind. Engr. Conf., 1985.

[RR] Reddi, S. S. and Ramamoorthy, C. V. "On the flow-shop sequencing problem with no wait in process," Operational Research Quarterly 23 (1972), 323-331.

[SC] Shen, V. Y. and Chen, Y. E. "A scheduling strategy for the flow-shop problem in a system with two classes of processors," Proc. 6th Annual Conf. Inform. Sys. Sci., Princeton Univ., Princeton, NJ, 1972, 645-649.

[Tr] Trivedi, K. S. Probability and Statistics with Reliability, Queuing, and Computer Science Applications, Prentice-Hall, NJ, 1982.

MINIMAL TECHNOLOGY ROUTING AND SCHEDULING SYSTEMS BASED ON SPACE FILLING CURVES

L. Platzman and J. Bartholdi
School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

Our research over the past few years has been directed toward the transportation and storage systems that link together the elements of any large manufacturing facility. We have developed a family of heuristic control algorithms that consist essentially of maintaining sorted lists of tasks to be performed. The lists are easily maintained and modified: New tasks may be added to a list, or tasks previously in the list may be removed, without requiring the entire list to be recomputed. These algorithms are thus ideally suited to real-time decision-making.

The sophistication of these procedures is embedded in the criterion with respect to which tasks are sorted. These critera may in some cases be expressed analytically as a "spacefilling curve." In others, it must be computed off-line, as a "presequence" - the solution to a large and difficult integer program. However, the criterion is designed only once, before the algorithm goes on-line. After that, the only computational operations are (1) evaluating the criterion for a particular task, and (2) appending or removing tasks from sorted lists.

The criterion is determined by first specifying a metric on the set of all possible tasks, which expresses the ease or efficiency of following one task by another. The spacefilling curve or presequence then maps the unit interval continuously onto the set of all tasks. Positions along the interval determine the criterion for sorting. Since adjacent tasks in a queue have nearly equal criteria, it is efficient to perform them consecutively.

A nontechnical introduction to our method may be found in

> Bartholdi et al., "A Minimal Technology Routing System for Meals on Wheels," Interfaces 13:3 June 1983, pp. 1-8

wherein we describe how an implementation of the procedure on a pair of Rolodex files enabled a charitable organization to efficiently solve a large routing problem (delivering lunch to more than 200 delivery points whose locations change daily) without any computer!

A broader survey of applications (including manufacturing) and underlying theory may be found on a paper to appear in a forthcoming Management Science special issue devoted to Heuristics. Copies of this paper (and others in progress) may be obtained by writing directly to the authors.

A MULTI-PASS EXPERT CONTROL SYSTEM (MPECS) FOR FLEXIBLE MANUFACTURING SYSTEMS

Professor Richard A. Wysk
Szu-Yung (David) Wu
Neng-Shu (Bob) Yang

207 Hammond Building
University Park, PA 16802

Department of Industrial and Management Systems Engineering

The Pennsylvania State University

INTRODUCTION

Since the industrial revolution some 80 years ago, mathematicians, engineers, scientists and production managers have been trying to develop efficient factory scheduling/control procedures. Over the past 80 years, many conflicting results concerning rules, procedures, performance measures and analyses have been reported in the literature. In general, mathematicians and operations researchers have tried to resolve the problem optimally and have discovered that optimal analysis is very difficult. On the other hand, production managers and production engineers have tried to use heuristic procedures to order and organize production flow. Unfortunately, the results from these methodologies are confusing and seem to be dependent on manufacturing system detail so that no general policies have been developed. Statistical validation of these analyses is also very difficult.

The advent of Numerical Control (NC) and Flexible Manufacturing Systems (FMS's) have highlighted our inability to effectively schedule machines automatically. In an FMS, the system controller is responsible for making scheduling decisions. Simple decision rules can alter the system output by 30% or more [1]. Selecting the proper scheduling rules in flexible automated systems is as difficult as it is in conventional manufacturing systems; however, unlike conventional systems, the rules must be explicitly defined.

In recent years, the theory of scheduling has been enhanced significantly from a Mathematical Classification stand point (see, for example, reference [3] or [7]). Factory scheduling problems have been shown to be in the class of problems called "NP-hard". A characteristic of

this class of problems in that solution time increase exponentially as problem size increases. Despite many years of research effort devoted to these problems by mathematicians and engineers, no solution procedures have been found that do not possess this characteristic of exponential growth. This has led many researchers to the conjecture that no solution procedure exist. For example, if 8 jobs are to be scheduled and each job requires 4 operations, then the number of possible semi-active schedules that must be examined would be $(81)^4$ or 2.65×10^{18}. The job routing requirements would render some of these schedules infeasible, but the number of schedules can still be very large (perhaps half the theoretical number). Unfortunately only very small scheduling problems (2 or 3 parts on 2 or 3 machines) can consistently be optimally resolved.

The growing interest in artificial intelligence and expert systems has led many computer scientists to advocate applying these methods to scheduling problems. An expert system can be defined as a "Tool which has the capability to understand problem specific knowledge and use the domain knowledge intelligently to suggest alternate paths of actions." The misconception about Expert Systems is due to considering it as a collection of IF..THEN rules along with some method of transferring knowledge from experts to non-experts. Expert systems not only use techniques to transfer knowledge but also use analytical tools to evaluate the knowledge and techniques to learn. A program having a few IF..THEN constructs and a set of analytical tools cannot be classified as an Expert System. An Expert System should be able to understand new knowledge, draw inferences, justify and explain its reasoning process [26].

Expertise in a domain is directly related to the knowledge that the expert has. By the same reasoning, the efficiency of an Expert System depends on the available knowledge. Hence Expert Systems are also known as Knowledge Based Systems (KBS). General AI methods attempt to solve generalized problems, the so called weak methods of AI, do not have sufficient expertise to solve problems effectively. Since each Expert System is tailored to a specific domain, the degree of reasoning and related strategies in these systems is much greater than general AI problem solving methods.

One approach to resolving factory scheduling problems to show promise recently is a procedure called "Multi-Pass Scheduling." Both Dar-El and Wysk [2] and Nof and Gurecki [5] have shown that significant throughput improvements can be made by using a simulation model to determine the future course for a manufacturing system. Essentially, the procedure works as follows. A simulation model of the system is resident on the system control computer. At each decision point, a deterministic simulation is run to see what control policy (from a series of rule based policies) impacts the current system most favorably. This control is then chosen and the appropriate control response is signaled for execution on the system.

One unfortunate drawback of this procedure is that FMS's are not static (the types of parts, demand, tooling, etc., change over time).

Each time these specifics change, the simulation model must be rewritten or updated which would have to be done manually. Although the benefits can be significant, the implementation cost can be quite expensive. However, the necessary data for both controlling as well as simulating the manufacturing system must be available to the system controller.

In this paper, an approach to utilizing a "Multi-Pass Expert Control System" (MPECS) for manufacturing cell control will be presented. Key elements of the system include:

1. An Expert System to select potential scheduling alternatives which form an alternative space.

2. A simulation model that is automatically generated by the control system.

3. A structure to allow the system to simulate system performance based on the alternative space, i.e. use the simulation model as a source of feedback for system decision making.

4. A decision structure that will update performance rules based on "simulation/system experience".

5. A mechanism to affect the control on a variety of Flexible Machining Cells (FMS's).

The basic principle behind the MPECS system is using deterministic simulation as a short term predictive tool for alternative control strategies in a manufacturing cell. This concept is illustrated in Figure 1. As can be seen from the figure, there are many controllable (endogenous) variables that must be affected. There are also many uncontrollable (exogenous) factors that will impact the system. A deterministic simulation can be run as a Gantt like scheduler to analyze short term effects. Decisions based on interference, machine utilization, etc. can be accessed for short term sequencing and operational decisions. This allows for use of a reasonably simple set of rules that are evaluated to determine performance.

The simulation does not examine exogenous factors, MPECS will respond to exogenous factors by creating a new simulation experiment/ model when unforeseeable events occurs. New rules may result when these conditions occur. However, the scheduler need only respond to these events rather than try to minimize conflict from them.

In the following sections, a complete description of MPECS will be presented. Each module of MPECS will be described in detail using examples.

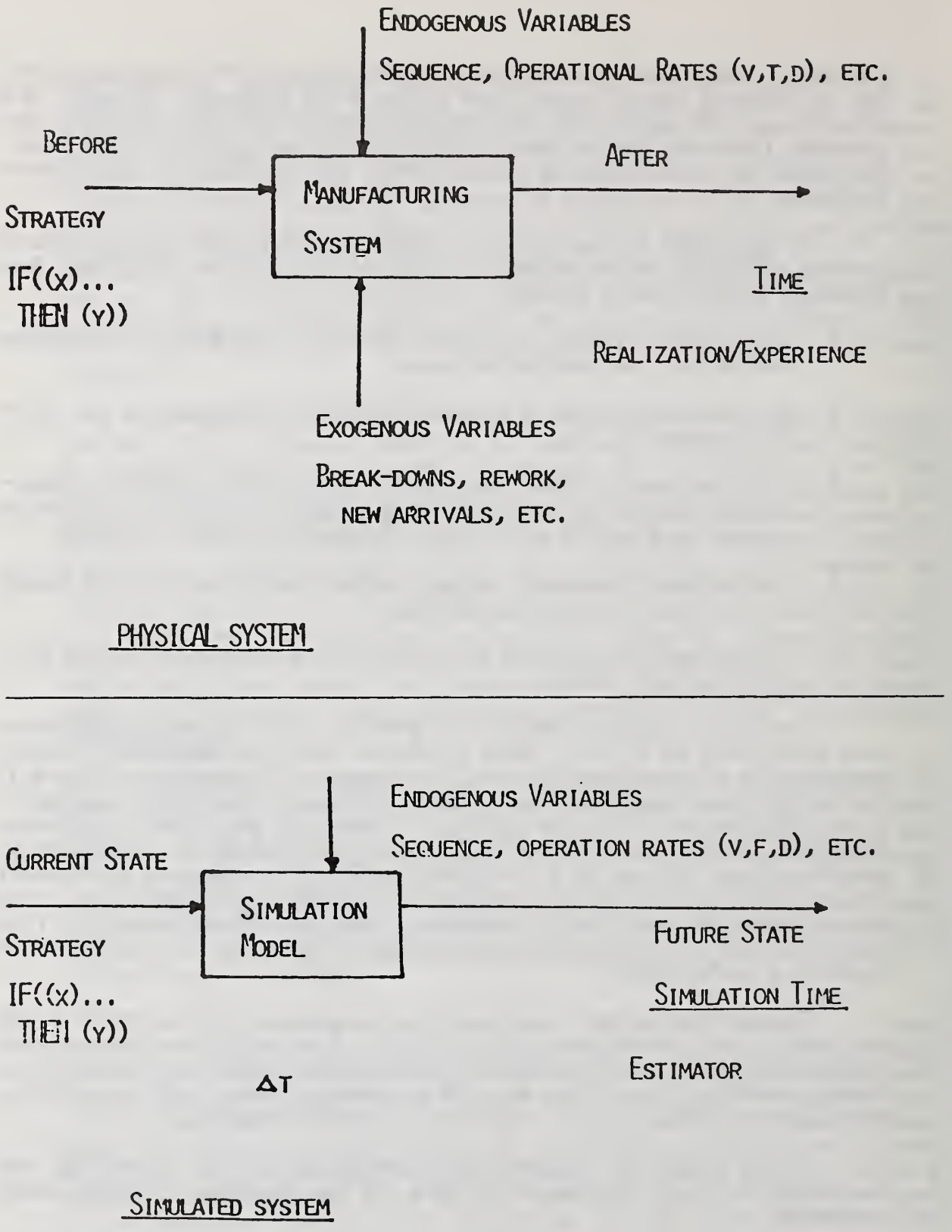


FIGURE 1. BASIS OF MPECS

GENERAL STRUCTURE OF THE CELL CONTROL SYSTEM

AN OVERVIEW OF THE SYSTEM

MPECS is a modular stand-alone control system which may be adapted into various kinds of hierarchical control system components with minor changes. The main purpose of MPECS is to: 1) utilize all the data available in a computerized manufacturing cell, 2) create "good" strategies to guide the system, and 3) generate real time responses to make control decisions during system run-time. As can be seen from Figure 2, operational information such as part routing specifications, machine loading information and material requirements are supplied by the factory control system. The managerial objectives such as due date, cycle time and cost constraints are also input to MPECS. This data provides factory information and constraints as well as objectives for MPECS. MPECS also supplies feedback to the factory control system. This feedback may include the information such as : a certain order is finished, additional material is required, a certain order can not be made by a specific due date,...,etc.

The general scheme of MPECS is pictured in Figure 3, MPECS consists of three major components: 1) an intelligent scheduling module (ISM), 2) a simulator and, 3) an actual cell control module (CCM). The implementation of the MPECS is partitioned into a decision and operational level activities. Decision level activities can be described as follows: Upon receiving a job order from the factory control system, ISM is activated to evaluate its knowledge base which contains scheduling rules and principles as well as shop floor information (e.g. three of 50 parts types are now been processed in the cell, status of the machines, etc.). After applying a series of inference procedures, ISM will eventually generate an alternative space which contains several "good" alternative scheduling rules. The alternative space will then update a working rule module (WRM) which contains dispatching rules and scheduling heuristics suggested by ISM.

For the operation level activities, the WRM is input to the simulator for further performance evaluation. Under regular operating conditions, whenever a request to dispatch a finish part from a idle machine is issued, the simulator is activated to evaluate the alternatives in the WRM. Based on the current system criteria and the future master production schedule of the shop, the simulator will run a simulation model to evaluate the alternatives in the WRM. Eventually, the "best" scheduling rule is selected from the WRM. The CCM then receives an execution command , which is generated based on the scheduling rule, to actually move (or wait) a part(s) in the system.

The ISM may be activated to make decisions under various conditions. As in the previous case, the "arrival of new job orders" can activate ISM's decision making. Some other conditions such as a machine break-down or a change of managerial objectives (e.g. expediting a certain job type is desired, etc.) can also activate ISM.

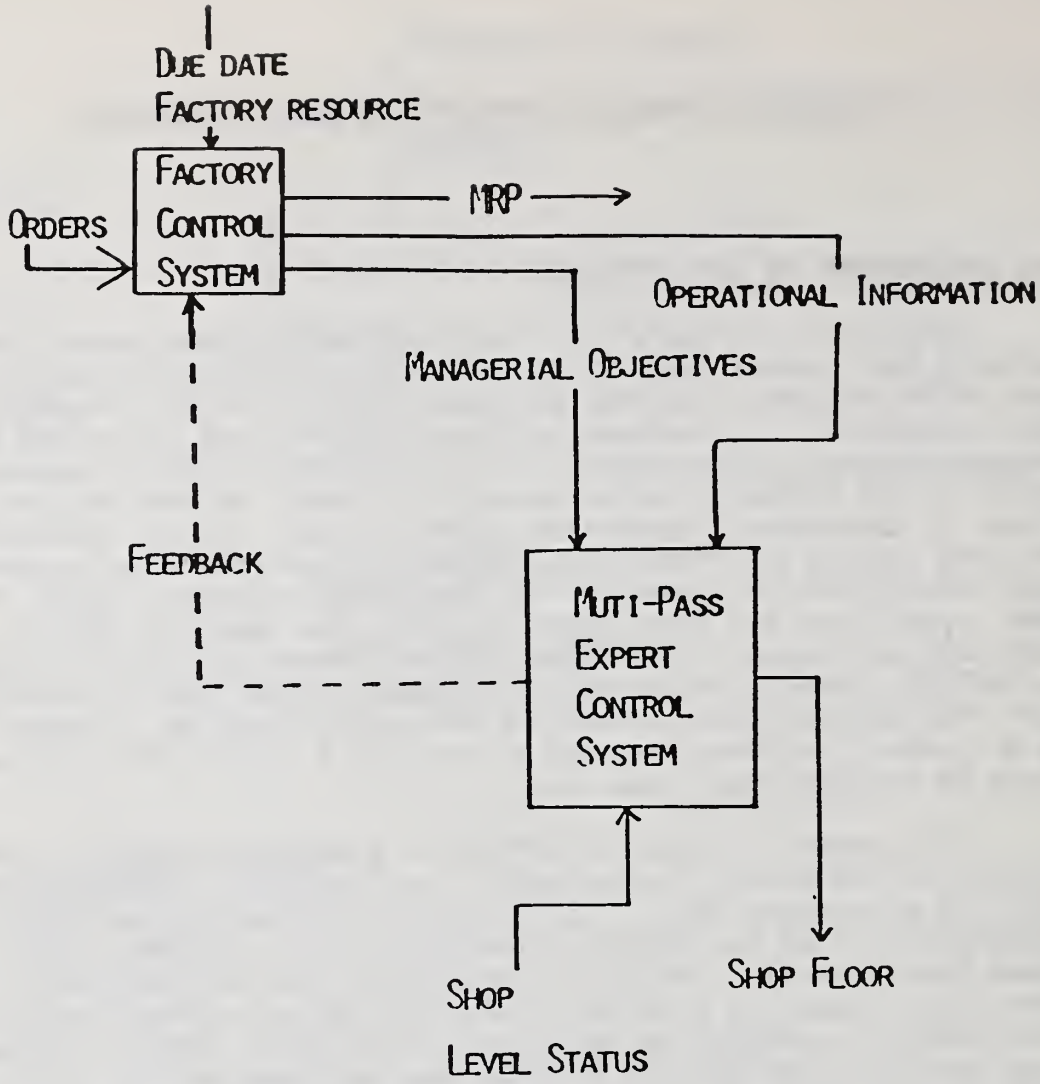


FIGURE 2. OVERVIEW OF MPECS.

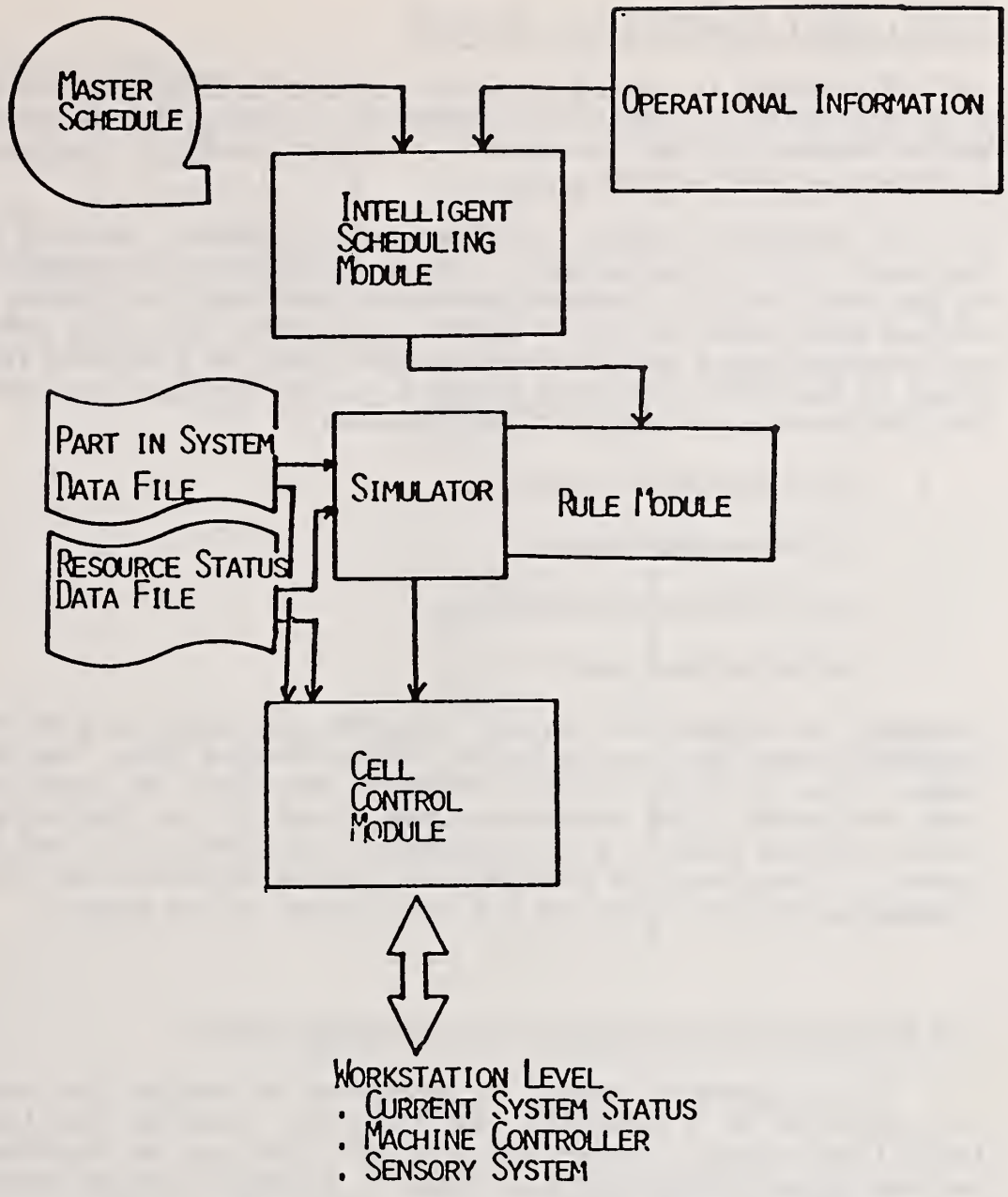


FIGURE 3. GENERAL SCHEME OF THE MPECS

INTELLIGENT SCHEDULING MODULE

Scheduling is one of the most important MPECS functions. Cell level scheduling is a principle vehicle for: utilizing the resources efficiently, responding the managerial objectives rapidly, and satisfying the system constraints effectively.

A real-time, flexible, and intelligent scheduling module (ISM) is the key to any control system. The scheduling module enables MPECS to generate various schedules based on input from the factory control system along with the actual status at the shop floor. ISM utilizes expert methodologies to assist in shop scheduling. A functional representation of the ISM is shown in Figure 4. As can be seen from the figure, the ISM consists of four major components:

1. the acquisition module,
2. the knowledge base,
3. the inference engine, and
4. the alternative space.

Basically, the acquisition module transfers knowledge and expertise of scheduling into facts and rules for the knowledge base. The inference engine then employs various inference strategies to manipulate the rules and facts in the knowledge base. Eventually, an alternative space which contains several "good" alternative scheduling rules will be generated. A simulation of the cell then further evaluates the alternative scheduling rules to determine the performance of the system.

The Acquisition Module And The Knowledge Base

The acquisition module is employed to transfer the knowledge and expertise of a scheduler into facts and rules for the knowledge base. This module is constructed so that ISM can be continually expanded. Since scheduling knowledge is primarily domain-specific, (i.e. no generally applicable scheduling rule is available for the various shop floor environments), the knowledge base of ISM is continually expanded to include various scheduling environments of different machine/part configurations with different levels of complexity. In addition, the acquisition module employs machine learning techniques to "learn" from feedback from the shop floor. In other words, the acquisition module attains knowledge from actual experience on the shop floor.

The knowledge base is a database which describes facts and rules associated with the scheduling problem domain and represents the facts and rules in clausal form of first order predicate logic. For example,

```
(forall (x) (implies
                (and (in-machine-queue x)
                     (shortest-process-time x))
```

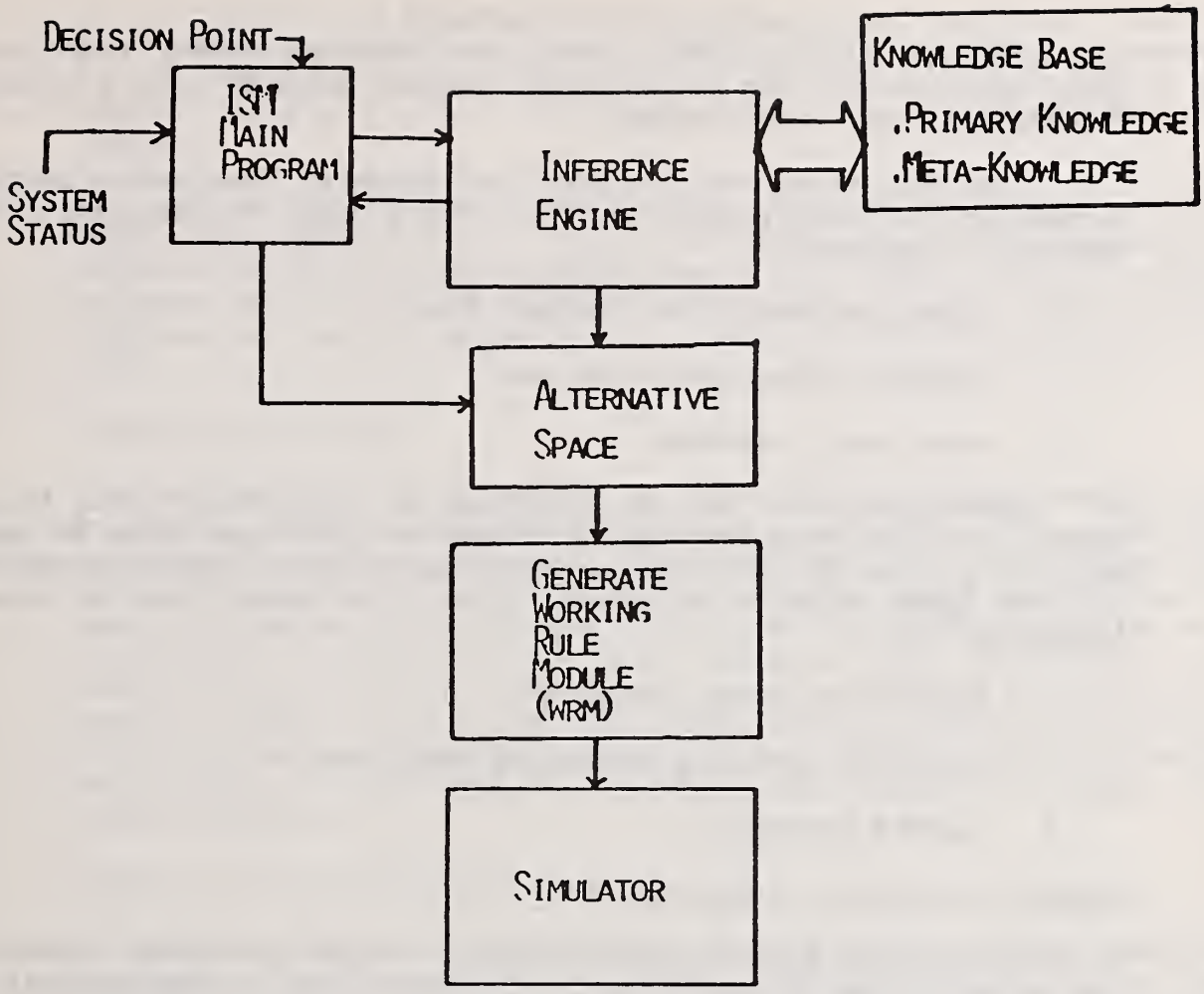


FIGURE 4. THE INTELLIGENT SCHEDULING MODULE (ISM)

(set-highest-priority-to x)))
can represent dynamic SPT rule (i.e. if x is a part waiting in the machine queue, and x has the shortest imminent process time, x is given the highest dispatching priority.)

The knowledge base contains two levels of information: primary knowledge and meta-knowledge. The primary knowledge consists of the following components:

1. status information on the shop floor,
2. dynamic dispatching rules, and
3. scheduling heuristics

Meta-knowledge can be described as "knowledge about knowledge". In ISM, meta-knowledge consists of the knowledge to apply scheduling rules and heuristics to previously inexperienced circumstances. The three areas of knowledge in the ISM categorized as meta-knowledge are:

1. general dominant principles,
2. criteria of selecting scheduling rules, and
3. learned heuristics

Status information on the shop floor

In the knowledge base, status information of the shop floor represents the current status of each system component (e.g. current process on each machine and material handler, etc.). The status information is supplied by the shop floor sensory system and workstation level controllers through the CCM. After receiving status information from the CCM, the acquisition module of ISM translates the information into assertions of fact. For example, the status of a certain machine may be represented as:

```
(mach-status
      (machine mach-11)
      (current-op job-2110)
      (proc-time 16)
      (time-remaining 5)
            .
            .          )
```

This information would be pre-processed to MPECS by the CCM.

Dynamic dispatching rules

Scheduling decisions in a real-time cell control environment are usually determined using dynamic dispatching rules. Both local and global rules can be implemented in a scheduling system. A rule is local if priority assignment is based only on information concerning the jobs represented at the individual machine queue (e.g. SPT rule).

A global rule utilizes information from other machines in addition to the individual machine queue. A good example of global dispatching rule is WINQ (i.e. work in next queue; job with the least work in the next machine queue has the highest priority in the current machine queue).

Several simulation studies in dynamic dispatching rules are available in the open literature. These rules are feasible for different domain applications. Different assumptions and environments tend to produce different results in using dynamic dispatching. Dynamic rules can be employed in the knowledge base as the basic scheduling reference of ISM.

Scheduling heuristics

Various scheduling heuristics can be employed in the knowledge base. In general, scheduling heuristics are more suitable for rather complicated system situations. Basically, the scheduling heuristics search and generate one or more schedule(s) to guarantee better system performance. Because heuristics can satisfy more detailed constraints in a complicated system, they are more practical than dispatching rules. Some heuristics however can become very complicated and too inefficient for application in a real-time environment. Hence the knowledge in this component is applied only when no dynamic dispatching rule can be applied or more delicate schedules are essential to satisfy the system constraints.

General dominant principles

For various objective functions and system configurations, dominant scheduling rules/procedures may exist. Dominant principles are facts and phenomena which can be proved to be true under specific conditions. Many times dominant principles can significantly reduce the size of a scheduling problem.

Examples of the dominant principles are listed as follows:

A. IF several machines are identically tooled and
are capable of performing the same operation
THEN logically pooling these machines into
a machine group will reduce the complexity
of the sequencing problem

B. IF an operation can be performed on any of the several
machines
THEN find a machine that is free

The dominant principles shown above are both intuitive and supported by various simulation studies. These dominant principles can drastically improve the performance of the schedule. For instance, based on principle A, the waiting time in the machine queues will decrease significantly. Meanwhile, the system will be better balanced because of the flexibility gained. Obviously, the more dominant principles that can be applied, the more efficient the scheduling process.

In the knowledge base, dominant principles served as meta-knowledge. From the previous discussion: applying dominant principles to guide the scheduling process may significantly reduce the complexity of the problem. Hence this level of meta-knowledge should be applied first in the inference procedures.

Criteria for selecting scheduling rules

In general, the criteria for scheduling is based on managerial objectives as well as the nature of the shop (e.g. level of workload, existence of assembly operations, etc.). Managerial objectives are primarily related to the information of processing time, due date, arrival time, costs, set up time and machine attributes. These objectives include: minimize throughput, minimize tardiness, maximize machine utilization, etc.

For certain managerial objectives, simple dynamic dispatching rules or scheduling heuristics may be sufficient to obtain a good performance. For other objectives, combinations or weighted combinations of simple rules are necessary to satisfy more complicated objectives.

Criteria for selecting scheduling rules are utilized in the knowledge base as meta-knowledge. Meta-rules (constructed from the meta-knowledge) are represented as production rules to assist in the reasoning process of the inference engine. For example, a meta-rule for maximizing throughput in a machine shop can be expressed as follow:

IF no assembly operations exist
  AND no due date constraints are active
THEN SPT rule should be considered

Learned heuristics

As mentioned earlier, the acquisition module is capable of learning from system feedback. Feedback is treated as actual experience gained from current scheduling activities. Experience which should be recorded by the knowledge base includes the following:

1. mistakes made by ISM,

2. successful schedules for typical system decisions, and

3. discovered dominant principles.

Learned heuristics are extremely important in a real-time control environment. Since this area of knowledge "remembers" mistakes which were made, the ISM can prevent making the same mistake for similar circumstances in the future. At the same time, ISM "remembers" some typical successful schedules for some system states, if a repeated state shows up, without going through the whole scheduling process ISM will generate an identical schedule. Furthermore, some generalized dominant principles can be generated based on the learned heuristics.

The Inference Engine

An important characteristic of expert systems is the distinction between the knowledge and the reasoning mechanism. Unlike the knowledge base, the reasoning mechanism, or inference engine, is not domain specific. The inference engine employs various reasoning techniques, such as forward chaining and backward chaining to manipulate the knowledge base.

In order for a mechanism to reason, it must be able to infer new facts from what it has been told. This involves dynamically creating a new symbol structure from the old ones. In the inference engine of the ISM, new rules are produced by applying rules in the knowledge base.

The inference engine of the ISM is basically a production system which consists of three basic modules: 1) a matching module, 2) an inference dependency module and 3) an execution module. Two types of production rules are employed in ISM -- forward chaining and backward chaining rules. Forward chaining rules are defined as

(... (IMPLIES (antecedent)(consequence)))
(i.e. if the antecedent(s) is true (matched) then
the consequence(s) is implied to be true)

While the backward chaining rules are defined as

(... (IMPLIES-B (antecedent)(consequence)))
(i.e. in order to prove the consequence(s) is true
prove that the antecedent(s) is true)

In general, the forward chaining rules are antecedent-driven (i.e matching is based on antecedent) while the backward rules are consequence-driven (i.e. matching is based on consequence). In applying an assertion or a query to the inference engine, the matching module will search for the rules in the knowledge base whose conditions (i.e. antecedent or consequence) are matched. The inference dependency module will then decide how to choose which rule to execute, and in what sequence. The determination of the inference dependency is based on both forward and backward chaining inference strategies. Finally, the execution module will execute
the chosen rules. The result of this execution is a modification of the knowledge base (i.e. generating new rules, insert new facts etc.) or an answer to the query.

The following is a simple example to explain a backward chaining inference procedure. Suppose the following rules and facts are exist in the knowledge base,

```
Rule 1:
  (FORALL (x)
    (IMPLIES-B
      (AND (x's utilization is maximized)
           (x is a machine)) ...(i.e. antecedents)
      (x's idle time is minimized)))...(i.e. consequence)

Rule 2:
  (FORALL (x)
```

(IMPLIES-B
(AND (WINQ rule is applied in the system)
(x is a machine))
(x's utilization is maximized)))

Rule 3:
(FORALL (x)
(IMPLIES-B
(AND (WINQ rule is applied in the system)
(x is a machine))
(x's idle time is minimized)))

Fact 1: (mach-11 is a machine)

Fact 2: (mach-12 is a machine)

Fact 3: (WINQ rule is applied in the system)

The goal is to check if mach-11's idle time is currently minimized. First, a query in the following form is issued.
(query: mach-11's idle time is minimized)

(i.e. issue a query to find out if
mach-11's idle time is minimized)
Since no fact concerning idle time exists in the system, a backward chain must be issued to further evaluate the goal. After the backward chain is issued, the matching module will match the consequence of both Rule 1 and Rule 3 with the goal. The inference dependency module then has to decide which rule to be fired first. If Rule 1 is chosen, the execution module then fires (i.e. try to prove) the rule. Since its a backward chaining rule, in order to prove the consequence--
mach-11's idle time is minimized

two antecedents must be proved:
1. mach-11's utilization is maximized, and
2. mach-11 is a machine
Antecedent 2 is a fact exists in the knowledge base (i.e. Fact 1). To prove antecedent 1 is sufficient, antecedent 1 is set to be the subgoal at this point. The matching module is again activated to find that the consequence of Rule 2 (i.e x's utilization is maximized) can match the subgoal. Since only one rule is matched, the execution module is called directly. Similarly, the execution module will try to prove that the antecedent of Rule 2 (i.e. WINQ rule is applied in the system and mach-11 is a machine) is true. Since Fact 3 indicates that WINQ rule is currently applied in the system and Fact 2 says mach-11 is a machine, Rule 2 is proved successfully. Which means the answer to the query will be "true" and the goal is satisfied. Remember that Rule 3 is still waiting to be proved. However, since the goal is already satisfied by Rule 1 (i.e. via Rule 2), no further evaluation of Rule 3 is necessary.

The inference engine is the heart of the ISM. Although applying sophisticated inference strategies to accomplish the tasks like knowledge acquisition, abduction or learning require years of research, a

simplified mechanism with restricted constraints and assumptions can be constructed. Like the knowledge base, the inference engine can be continually modified by relaxing the constraints and assumptions one after the other.

The alternative space and the interface with the simulator

At each decision point, the ISM is activated to make a decision based on the present status of the cell. After a series of deduction and searching processes of the inference engine, the ISM will eventually evolve several alternatives which will form an alternative space for further decision making. The alternative space contains abstract information for feasible dispatching rules and scheduling heuristics selected by the system. Based on the abstract information, the ISM will generate a working rule module (WRM) for the simulator to evaluate all of the alternatives based on the future state of the system. As a result, the simulator will report the performance of different rules based on different interests of measures (e.g. cost, tardiness, makespan,...,etc.).

Since the simulator will further evaluate the alternatives, any mistakes made by ISM can be detected before actually apply to the physical system. Such mistakes may include: accumulating of queues, locking of the system or inefficiently utilize the machine, etc.

THE CELL CONTROL MODULE

The cell control module (CCM) of MPECS directly interfaces the workstation level with the control hierarchy. As can be seen from Figure 5, the basic tasks of the CCM are as follows:

1. Top down execution

   Upon receiving a command from ISM, CCM will execute the command in a top-down fashion. Such commands may look like

   Remove the finish part on machine-11 and put it in machine-queue-11, load part-1021 from machine-queue-2 to machine-11

   Such commands will be decomposed into a more detailed level in a step-by-step fashion. For the above example, the command may be decomposed to executive command for different workstation level system components as

   TO machine-controller-11: move the machining table to load/unload position

   TO material-handling-robot-1:
   1)execute motion sequence-153 (which will pick up the part from machine-11 and send it to machine-queue-11)
   2)execute motion sequence-213 [part-1021] (which

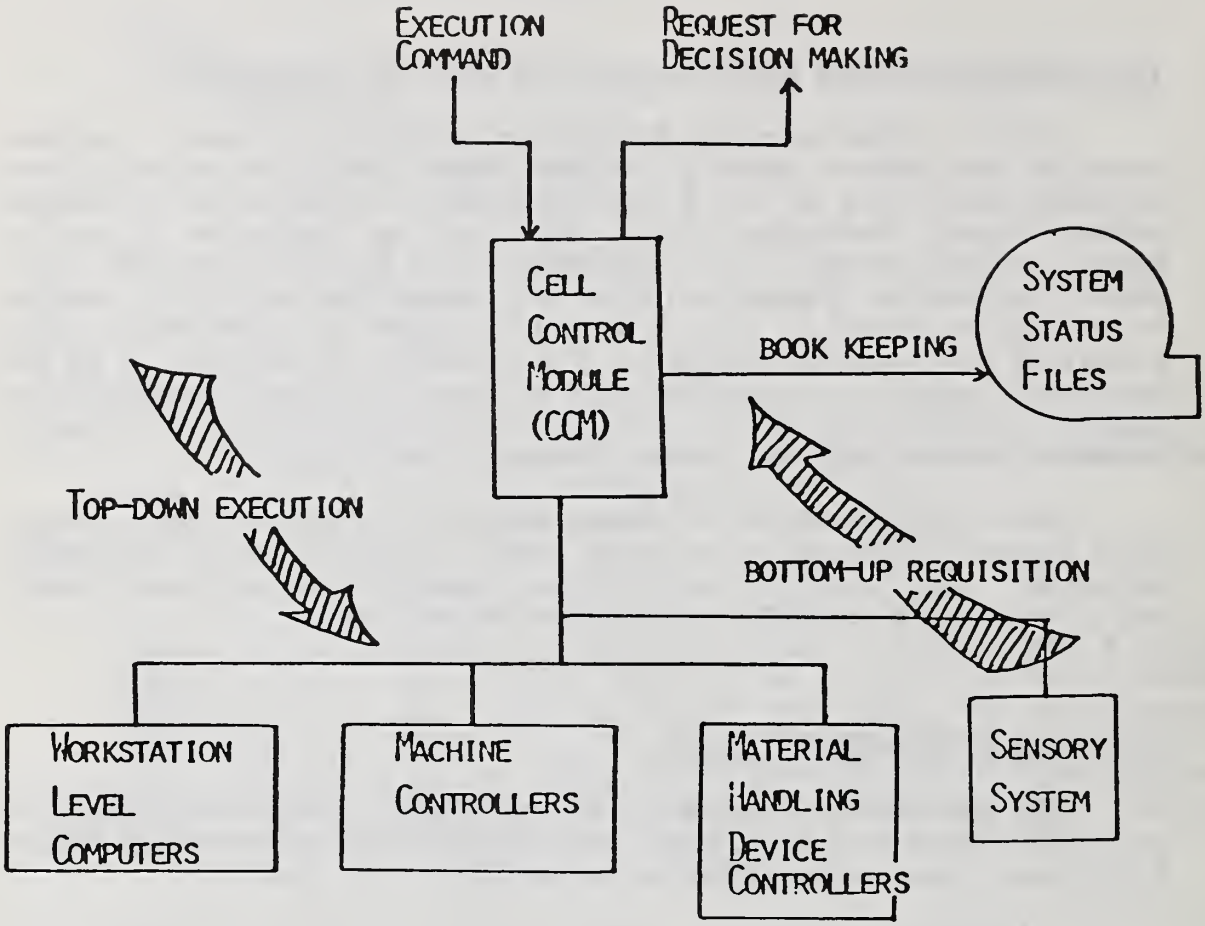


FIGURE 5. THE CELL CONTROL MODULE

will go to machine-queue-2 pick up part-1021
from there, send to machine-11)

2. Book-keeping

Whenever the system status changes, CCM will update the system status file which may include, a resource data file, part data file and miscellaneous system status file, etc. Furthermore, the CCM will report current system status to the knowledge base as discussed previously. In other words, the CCM takes responsibility of book-keeping for MPECS. The source of the system status information includes both the sensory system and all the lower level controllers within the workstation level. Actually, most of the shop floor information which is essential for cell control is obtained and manipulated by the CCM.

3. Bottom-up requisition

The CCM not only handles the top-down execution commands but also processes the requisition from the workstation level. For the situation in which direct requisition is necessary for the system, the CCM will receive the information from the workstation level in an interrupt manner. Such bottom-up information may includes: machine breaks down, system jams, a part is just finished,...,etc.

THE SIMULATION MODEL

The major function of the simulation model is to evaluate control policies in a flexible manufacturing system(FMS) by examining the effect of the production schedule on an on-line test base. The simulation model queries the part data and resource data files as input. It then determines the future system status by making a pass of deterministic simulation according to a rule(s) defined in the rule module. The system performance predicted by each pass of simulation is a measure of closeness to an objective function from the higher level factory control system. Thus at the end of all passes of simulation, the best schedule, resulting from the simulation, is then applied to the physical system.

INTERFACING THE SIMULATION MODEL

Given a set of parts, the inclusion of the simulation model in MPECS enables the control system to look ahead at the system performance among a certain number of alternatives. The simulation model is basically a mechanism which examines and records the performance of the system under different control policies.

The simulation model may be automatically activated by MPECS whenever a decision is to be made. Decision making may also be prompted by the arrival of new parts, the failure of resources, etc., in which case a production schedule needs to be created, or revised. Basically the simulation model receives a part data file and resource data file as input, and imitates the movement and processing of parts following the algorithm specified by the WRM. The system performance which is shown in a performance evaluator can then be compared to give the best production schedule. The interface of the simulation model is pictured in Figure 6.

Resource Data File

In an FMS, MPECS is installed in a physical manufacturing system which may include NC machines, material handling devices, machine queues, etc.. Therefore, MPECS can query the status of the FMS at any time and record the system history. The current system status is stored in a resource data file which is used as input to the simulation model. Various resource data files will be created during a pass of simulation so that MPECS is able to preview future system status.

Part Data File

When a set of parts are assigned to the FMS, MPECS also receives the necessary information to produce the part from the higher level factory control system. The information includes the detailed part program to process a part at each machine. Only some of the information is utilized by MPECS for creation of the simulation model. The part data file includes part routings, process times, due dates, etc. which are required to develop and execute the simulation model.

The Working Rule Module(WRM)

The WRM, which was discussed earlier, may contain dynamic dispatching rules or scheduling heuristics. Therefore, each pass of simulation follows a specific direction defined by a rule in the WRM. The WRM is integrated into the simulation model by MPECS at alternate decision points.

Performance Evaluator

Each pass of simulation yields a measure of performance associated with the control employed in the model. After all simulation passes have been performed, the best schedule can then be selected based on a system objective function. Therefore at each decision point, a perfor-

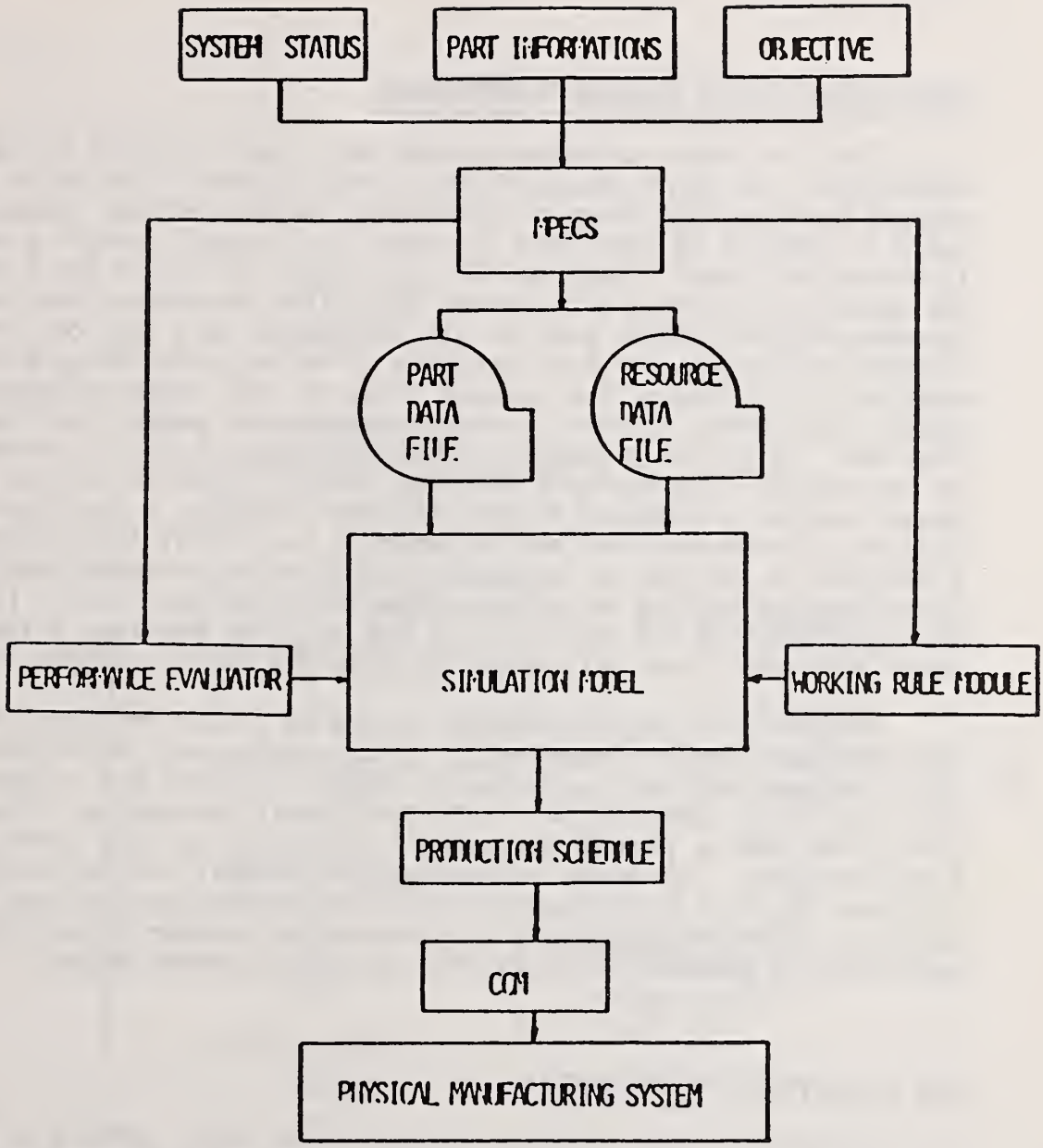


FIGURE 6. THE INTERFACE OF SIMULATION MODEL

mance evaluator is selected for the simulation model by MPECS based on the objective of the higher level factory control system. If the objective requires more than one measure of performance, then a weighting factor associated with the multiple performance evaluation is given to the simulation model.

THE SIMULATION MODEL SOFTWARE

The simulation software consists of a main program and a EVENT subroutine. An event calendar storing information is utilized in the exchange between the main program and the subroutine. Generally, an event is defined as that time in which the system status is changed. Therefore, an event could be the completion of a part on a machine, the arrival of a part at a machine, etc.. The discussion here will concentrate only on events such as the completion of a part on a machine in order to simplify the demonstration. The major function of the main program is to locate the nearest event in the event calendar. The EVENT subroutine is used to make appropriate actions for the event observed. After the completion of a part, some action is usually taken by the system. An example of these actions might be to put a completed part in a machine buffer, and then allocate a new part to the machine. These activities are depicted in the EVENT subroutine. Once a new part is chosen, its completion time on the machine can then be determined by adding its process time to the current time. Therefore, the completion of the new part on the machine becomes a new event which will replace the old current event in the event calendar.

Basically, the event calendar serves as a path through which the main program and the subroutine communicate with each other. The main program, which is pictured in Figure 7, reads the nearest event from the event calendar, and calls the EVENT subroutine. The EVENT subroutine makes proper decisions, and writes a new event on the event calendar. The basic structure of the EVENT subroutine is shown in Figure 8. The interaction between the main program and the subroutine continues until a certain termination criteria is met(completed operations or elapsed time), and the simulation model stops.

AN ILLUSTRATIVE EXAMPLE

Suppose a set of parts arrives at an FMS. MPECS is used to generate a production schedule which maximizes the resources utilization. The routings and process times for the parts are shown in Figure 9. Three very simplistic dispatching rules(shortest process time, least work remaining, and a random rule) are used to develop and illustrate the simulation model. The Gantt chart associated with each pass of simulation is pictured in Figure 10. As can be noted from the figure the difference between the best and the worst schedule is 35.8%. This however is one for a single rule not used in multi-pass application. Additional improvements might be possible.

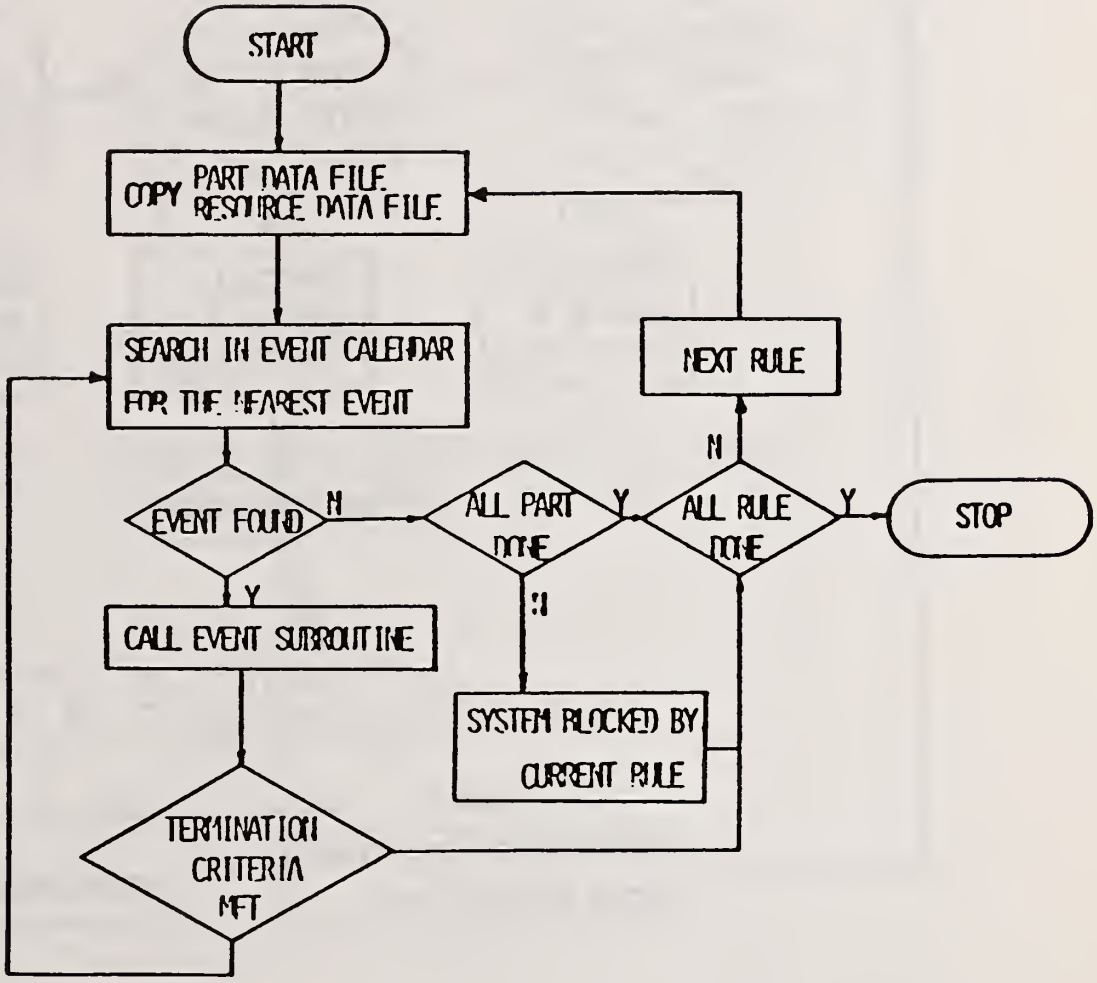


FIGURE 7. THE MAIN PROGRAM

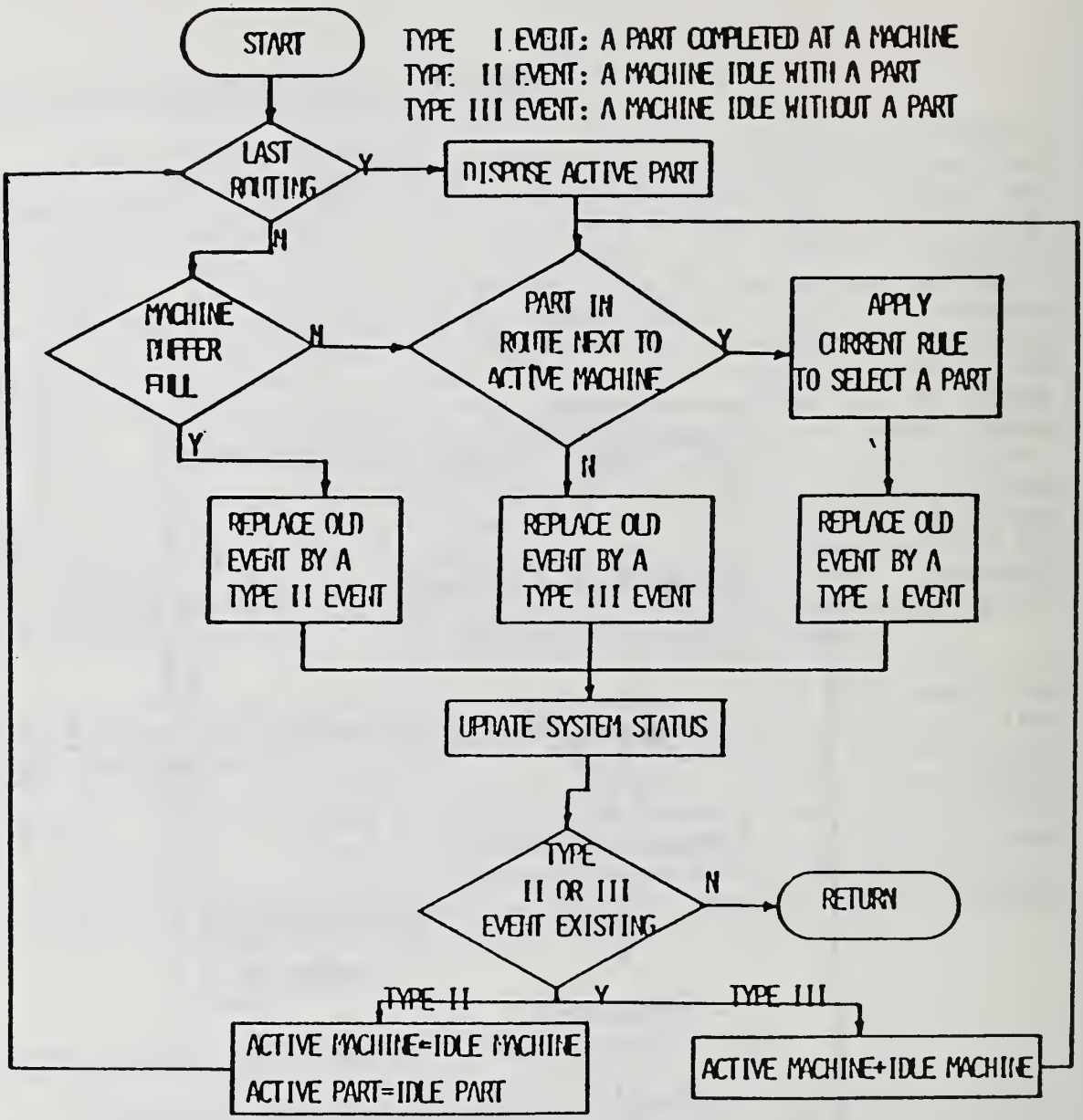


FIGURE 3. THE EVENT SUBROUTINE

PART NO.	ROUTING	PROCESS TIME
1	1-2-3	5-10- 5
2	1-3-2	10-12- 6
3	3-1-2	12- 6- 8
4	2-1-3	20-10- 5
5	2-3-1	10- 8-14
6	3-2-1	5-10-15
7	1-3	6-14
8	3-2	10-10
9	2-3	4-10
10	3-1	15- 5

FIGURE 9 PART ROUTINGS AND PROCESS TIMES

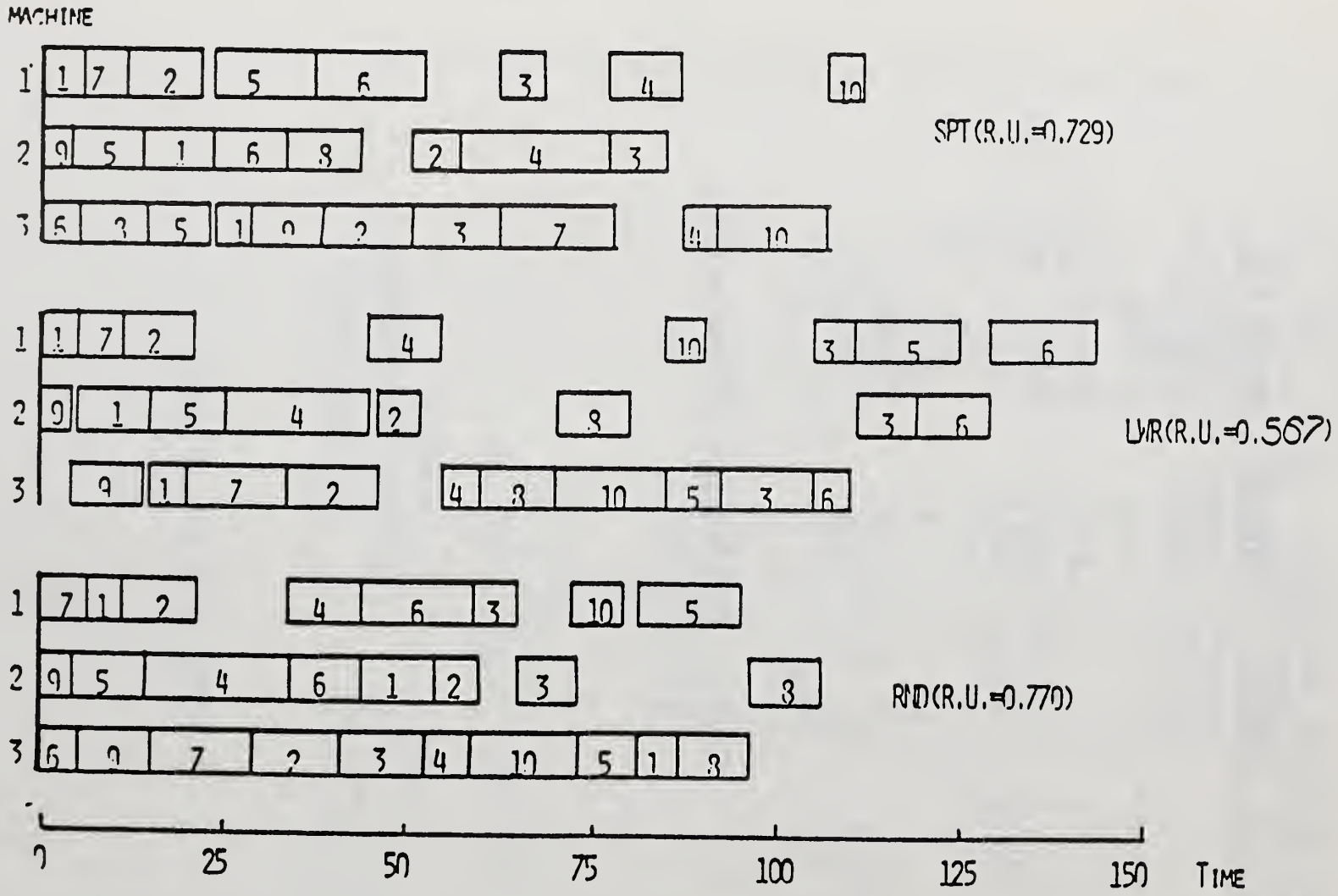


FIGURE 10. THE GANTT CHART OF SIMULATION RESULT

The simulation results depend on the parts mix, the objective and the time period in which all passes of simulation are performed. Therefore, off-line analyses are needed for the higher level factory control system to assign a group of parts, and for MPECS to determine a suitable time period to develop the analysis. The illustrated example shows that the random rule is better than the shortest process time for the objective of maximizing the resource utilization. However, if the simulation for the example was developed in two stages instead of one, and the first stage simulates 52 time units of activity, then the SPT rule(shortest process time) yields the best utilization of resources during that period. The random rule performs better during the next 52 time units. Furthermore, a different objective function from the higher level factory control system requires different criteria to control the simulations in order to produce the best possible production schedule. Therefore, the time window, which determines the length of the simulation in a stage, is a complicated function of the objective and the parts assigned. However the time window can be determined in an off-line analysis, and its on-line implementation is still feasible and efficient.

CONCLUSIONS

The organization and structure provides an efficient structure for FMS control. MPECS utilizes both AI and analysis procedures jointly to maximize the benefit attainable from both methods. Both the Expert System and Simulation/Analysis play a major role in the system. The Expert System provides the control for both the simulation and then the actual system. Although MPECS has not been fully developed, early results from the control modules appears to run efficiently enough to be included in even large production cells. The interface structure also seems to be easily implementable. Testing of MPECS should begin during the Summer 1986.

REFERENCES

1. Barash, M. M., et al., "The Optimal Planning of Computerized Manufacturing Systems", NSF Report APR74 25256, 1976.

2. Dar-El, E. M. and Wysk, R. A., "Job Shop Scheduling--A Systematic Approach", Jounal of Manufacturing Systems, Vol. 1, No. 1, 1982.

3. Aho, A. V., Hopcroft, J. E., and Ullman, J. D., The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, MA, 1974.

4. Matalon, E., "A Statistical Procedure to Determine the Economic Optimum in Computer Simulation", Upublished Masters Thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA, 1981.

5. Gurecki, R., and Nof, S. Y., "Decision Support for a Computerized Manufacturing System", The Optimum Planning of Computer Manufacturing Systems, NSF Report No. APR74-15256, No. 14, 1979.

6. Ignizio, J. P., "Solving Large Scheduling Problems by Minimizing Conflict", Simulation, March 1978.

7. Graham, R. L., "The Combinatorial Mathematics of Scheduling", Scientific American, Vol. 238, No. 3, March 1973.

8. Chang, T. C. and Wysk, R. A., An Introduction to Automated Process Planning Systems, Prentice Hall, Inc., Englewood Cliffs, NJ, 1985.

9. Garey, M. R., and Johnson, D. S., Computers and Intractability: A Guide to the Theory of NP-Completeness, W. F. Freeman and Company, San Francisco, CA. 1979.

10. Solberg, J. J., "Capacity Planning with a Stochastic Workflow Model", IIE Transaction, Vol. 13, No. 2, 1981.

11. Hildebrant, R., "Scheduling Flexible Manufacturing Centers When Machine are Prone to Failure", Ph.D. Thesis, MIT, May 1980.

12. Gallager, C. C. and Knoght, W. A., Group Technology, Butterworth, London, 1973.

13. Ham, I., "Group Technology," (chapter 7.8), the Handbook of Industrial Engineering, published by John Wiley Co., New York, 1982.

14. Ham, I., Hitami, K. and Yoshida, T., Group Technology Applications to Production Management, Published by Kluwen Publishing Co., Boston, 1985.

15. Descotte, Y. and Latombe, J. C., "GARI: A Problem Solver that Plans how to Machine Mechanical Parts," Proceedings of IJCAI-7, Vancouver, Canada, 1981.

16. Duda, R. O., Gashing, J., Hart, P. E., Konollge, K., Reboh, R., Barret, P., and Slocum, J., "Development of the PROSPECTOR Consultation System for Mineral Exploration," Final Report, SRI Project 5821 and 6415, SRI International Inc., Melno Park, CA, 1978.

17. Duda, R. O., Gaschnig, J. G., and Hart, P. E., "Model Design in the PROSPECTOR Consultant System for Mineral Exploration," Expert Systems in the Micro-Electronic Age, pp. 153-167, Edinburgh University Press, Ediburgh, 1979.

18. Dutta, D. and Joshi, S., "MR1: An Expert System for Configuration of Modular Robots," Some Prototype Examples for Expert Systems, ed. K. S. Fu, vol. II, pp. 282-316, Technical Report, TR-EE 85-1, School of Electrical Engineering, Purdue University, West Lafayette, 1985.

19. Eshel, G., Barash, M., and Chang, T. C., " A Rule Based System for Automatic Generation of Deep Drawing Process Outlines," Symposium of Computer - Aided/ Intelligent Process Planning, ASME Winter Meeting, Miami-Beach, Fl., Nov. 1985.

20. Faln, J., Gorlin, D., Hayes-Roth, F., Rosenshein, S. J., Sowizral, h., and Waterman, D. "The ROSIE Language Reference Manual," Technical Report N-1646-ARPA, Rand Corp., Santa Monica, CA, 1981.

21. Finin, T., McAdams, J., and Kleinosky, P., "FOREST: An Expert System for Automatic Test Equipment," Proceedings of IEEE Computer Society Conference on AI Applications, pp. 350-357, 1984.

22. Fisher, E., "FADES: Knowledge Based Facility Design," Ph.D. Thesis, Purdue University, West Lafayette, IN, 1984.

23. Forgy, C. L., "The OPS5 User's Manual," Technical Report CMU-CS-81-135, Computer Science Department, Carnegie Mellon University, Pittsbyrg, PA, 1981.

24. Fox, M. S., "The Intelligent Management System: An Overview," Technical Report CMU-RI-TR-81-4, Intelligent Systems Laboratory, The Robotics Institute, Carnegie Mellon University, 1981.

25. Fox, M. S., "Job Shop Scheduling: An Investigation into Constraint Directed Reasoning," Ph.D. Thesis, Carnegie Mellon University, 1983.

26. Kumara, S. R. T., Joshi, S., Kashyap, R. L., Moodie, C. L. and Chang, T. C., "Expert Systems in Industrial Engineering," IJPR, in process

27. Barr, A. and Feigenbaum, E. A., The Handbook of Artificial Intelligence, Vol. I, William Kaufmann, Inc., Los Altos, CA. 1981.

28. Barr, A. and Feigenbaum, E. A., The Handbook of Artificial Intelligence, Vol. II, William Kaufmann, Inc., Los Altos, CA. 1982.

29. Blazer, R., Erman, L. D., London, P., and Williams, C., "HEARSAY III A Domain Independent Framework for Expert Ssytems," Proceedings of AAAI, pp. 108-110, 1980.

30. Charniak, E. and McDermott, D., An Introduction of Artificial Intelligence, Addison Wesly, Reading, MA, 1985.

31. Cohen, P. R. and Feigenbaum, E. A., The Handbook of Artificial Intelligence, Vol III, William Kaufmann, Inc., Los Altos, CA, 1982.

REQUIREMENTS FOR AUTOMATIC CONTROL OF AEROSPACE MANUFACTURING PROCESSES

D.N. Pope Ph.D.
Manager, Simulation and Modeling
LTV Aerospace and Defense Company
Vought Aero Products Division MS 49R-32
P.O. Box 225907
Dallas, Tx. 75265

INTRODUCTION

In recent years, there has been considerable emphasis on the modernization of defense related industries. Manufacturing technology programs have advanced materials and process methods. Cost reduction incentive programs such as the Technology Modernization and Industrial Modernization Incentives Program (IMIP) have provided business arrangements that have encouraged capital investment in manufacturing resources. The US Air Force Integrated Computer-Aided Manufacturing (ICAM) program developed the overall operations architecture common to aerospace operations and funded technology programs directed at specific need areas.

The ICAM 1105 Factory of the Future project described the generic elements common to aerospace operations as illustrated in Figures 1 and 2. Figure 3 provides further characteristics of the aerospace environment, some of which are common to industry in general. In all manufacturing operations, and especially in aerospace, the bulk of the costs and challenges are in control of the processes rather than in the actual material transforming operations themselves. Only about 10% of total product costs involve direct shop floor labor, and part of that 10% are actually control activities. Current factory control typically consists of large manufacturing control organizations supported by data residing on central mainframes. Some machine processes are driven by controllers, perhaps linked to a DNC system for communication of NC part programs and status information. But overall, few devices are networked to each other or to the central mainframe and thus fail to acheive the desired level of control.

New technologies in process equipment, computer systems, product design, process planning, and decision support have potential for improving control from the management board room all the way down to the process control level. In this paper, the structure, requirements, and technology of control will be discussed, as well as some actual case studies.

A STRUCTURE FOR CONTROL

The sheer magnitude of aerospace operations has led to the decomposition of the overall environment into units, divided up on the basis of function, location, and product/program. In the past, this approach has not generally yielded global optimal

Figure 1. Generic ICAM Factory of the Future Concept.

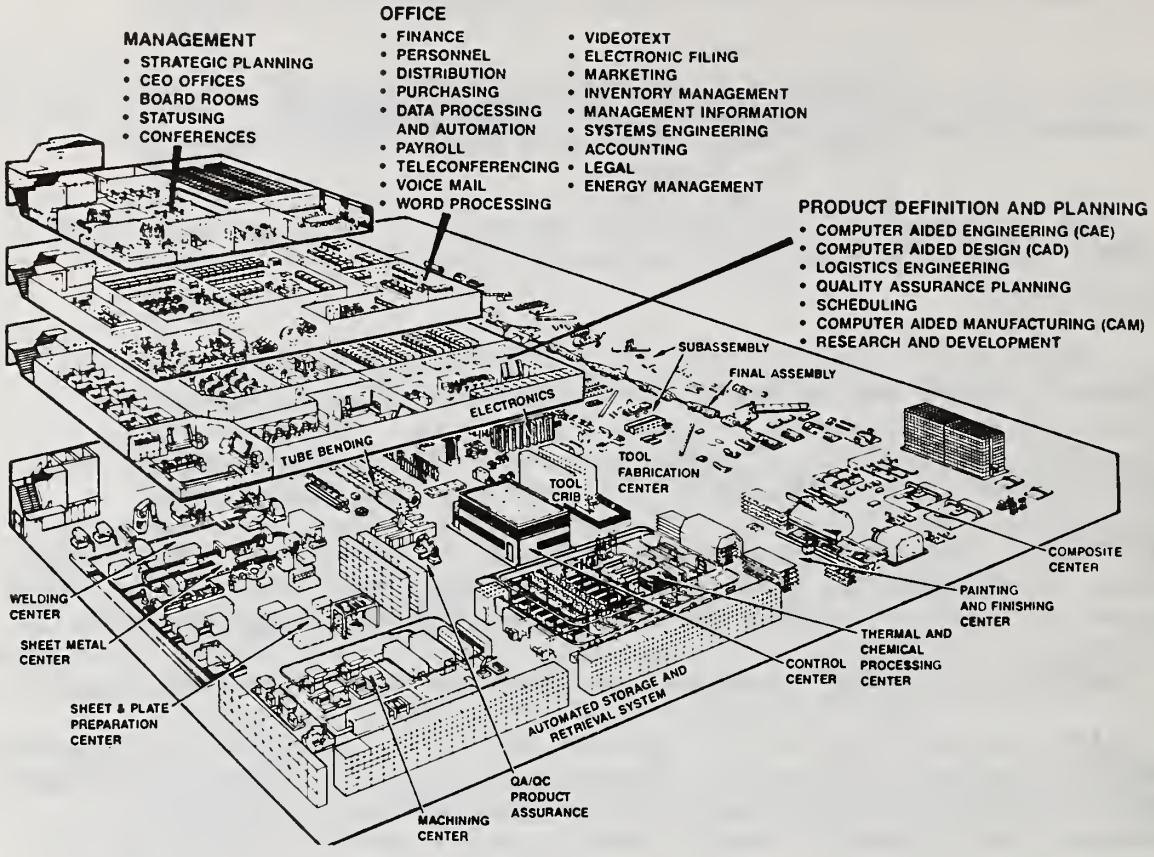


Figure 2. ICAM Factory of the Future Functional View.

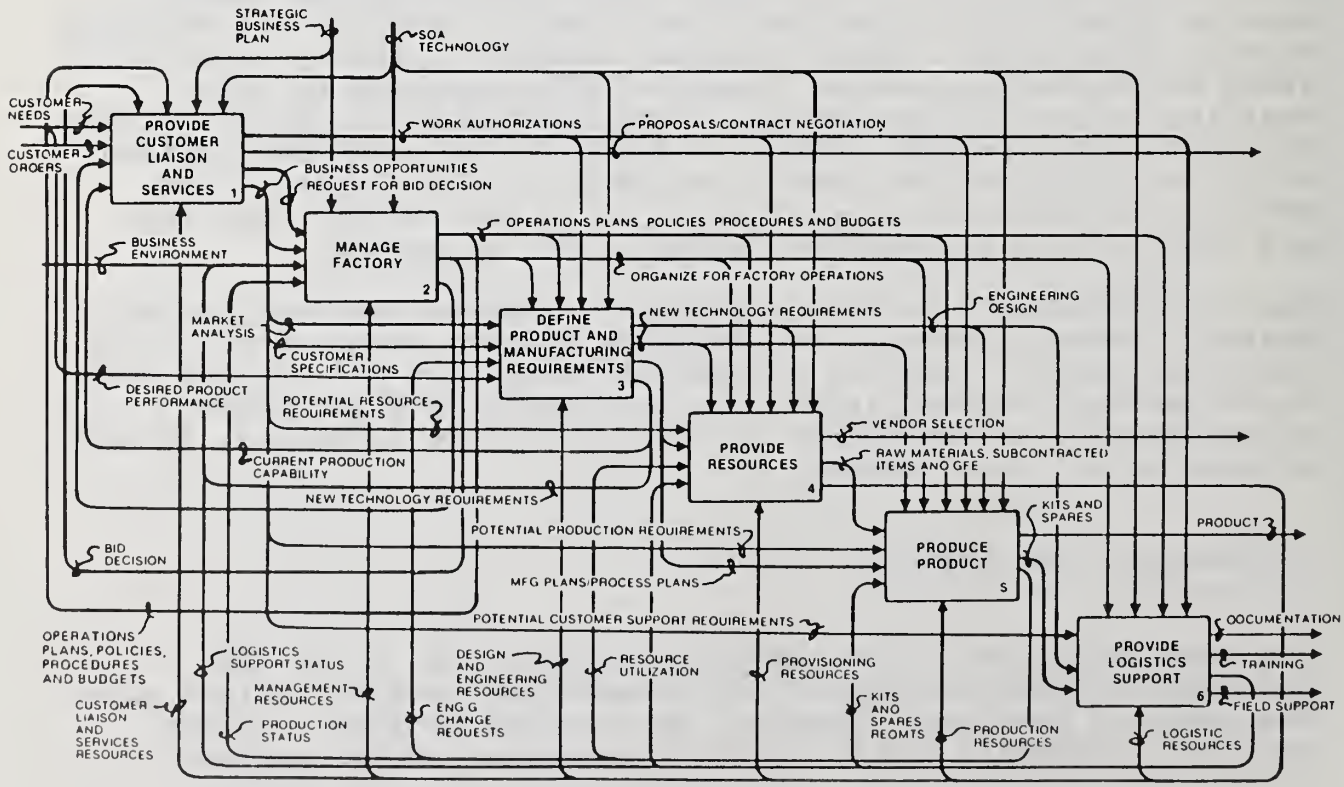


Figure 3. Characteristics of Aerospace Manufacturing.

Aircraft
---

* Low production rates (1-20) per month with extremely high part counts (30,000 - 100,000) per end item
* Detail fabrication and low level subassembly performed using small batches
* Span times of 1.5 years from detail fabrication through assembly and check out
* Current shop floor mechanization consists of flexible machining systems, robotic drilling, routing, fastening, composite tape laying, ultrasonic inspection
* Many facilities and techniques date back to World War II industrialization

Missiles
---

* Low to moderate rates (1-10,000) per month with high purchased component content
* Ordnance (rocket motors, warheads) handling environment in assembly operations
* Cleanliness requirements for component assembly
* Mechanization in electronic assembly and some structural assembly operations

Both aircraft and missiles
---

* Long product development time
* Exotic materials, rapidly evolving processes
* Rigid dimensional, process, and documentation requirements
* High engineering change volume
* Political and technological risks in long range production forecasting

results, partly because of lack of global data and partly because of lack of global management performance objectives. Hierarchical computer control schemes offer the mechanism to better meet the data requirements for optimal control and raise the performance measures of the subsystems to a more global level. Factory of the Future (FOF) architectures have been represented in various ways, but most identify levels of control such as:

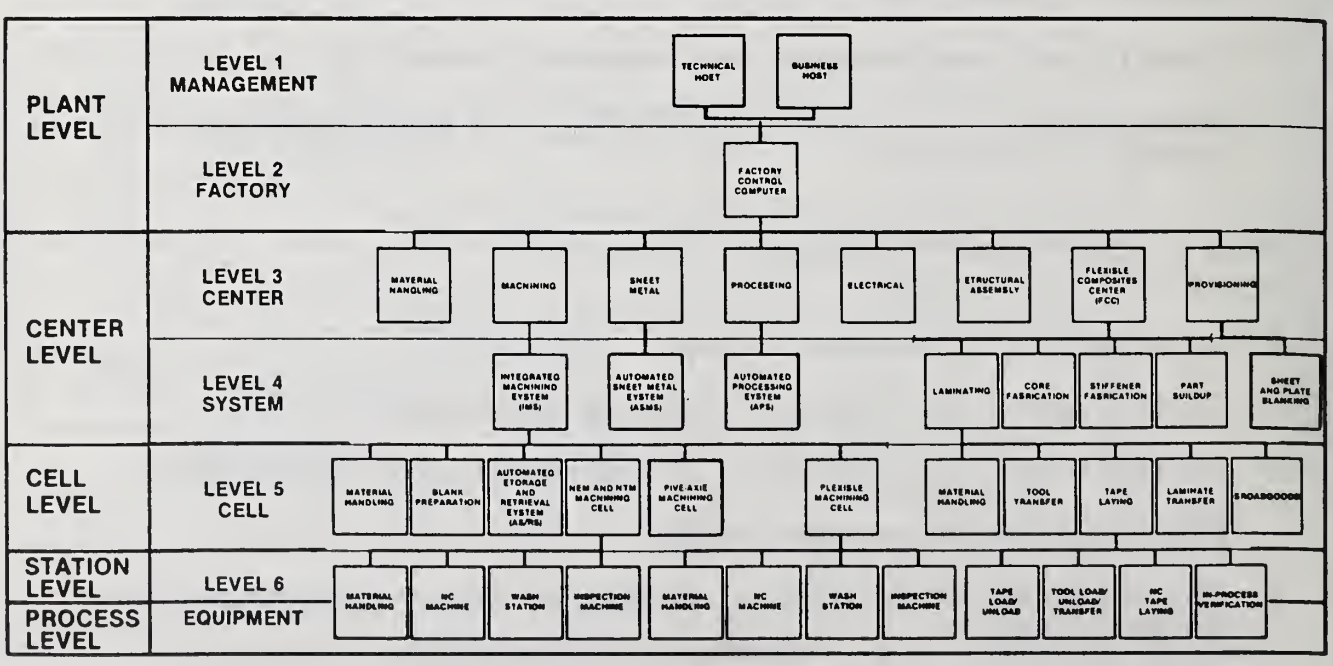
* Factory
* Center
* Cell
* Station
* Process.

The LTV Vought Aero Products Division (VAPD) FOF planning architecture, as presented by Harkrider (1), defines the hierarchy in this manner:

* Management
* Factory
* Center
* System
* Cell
* Equipment

as in Figure 4.

Figure 4. VAPD Hierarchical Computer Control Architecture.



A hierarchical, distributed control scheme offers the following benefits:

* The span of control at each level is manageable.

* Configuration control can be exercised on the global data required across many functions.

* Operation specific data can be managed at the level it is needed, in a timely manner, without burdening the entire system.

Successful implementation of the FOF control scheme is achieved by top-down design and bottom-up incremental implementation. Top-down design minimizes expensive retrofitting and islands of automation. On the other hand, bottom up implementation is a practical necessity. Industrial implementations of lower levels of control are now common place, i.e., adaptive control at the process level and cell control in FMSs. Sufficient bases of technology and experience now exist to drive the level of control up to the system and center levels.

The decomposition of control into major factory centers and cells follows a group technology classification of operations: assembly, sheet metal, machining, electrical, receiving, etc. Control modules are developed based on the following rationale:

* A control level is inserted only if there is more than one resource on the level below it.

* Control of a resource is performed at the lowest level having all the information required.

* Shared resources are placed under the lowest control module common to all modules sharing the resources.

* Control levels may be combined to exploit computer hardware efficiently. Position in the hierarchy does not indicate the extent or nature of computer hardware resources. Computer resources vary depending on the scope of activities and data at that level.

In order to control the shop floor in an optimal manner, more attention must be given to the management and scheduling of resources other than just the process equipment and material in work. Figure 5 summarizes some of the resources and activities which require control.

CONTROL TECHNOLOGIES

Numerous technologies are now available for implementing the hierarchical control strategy. One of the primary objectives of automation is direct labor reduction, but a benefit of equal

Figure 5. Resource Control Requirements.

Resources
---

* Information
  * Work order,schedule data
  * Process routings
  * Quality data
  * Product definition data
  * Supplier data
  * Specifications, Instructions
  * N/C part programs

* Material

* Equipment

* Tools
  * Cutters
  * Fixtures,Jigs
  * Supplies

* Personnel

Resource Management Activities

Information -

  Configuration - end item effectivity, etc.
  Storage and Communication

Physical -

  Transportation - move physically
  Location/status tracking
  Storage
  Build or procure resource
  Preparation - cleaning, etc.
  Maintenance
  Quality inspection and certification

importance is control. Thus all automation technologies are, in fact, means of achieving control over the manufacturing environment. Figure 6 summarizes the various automation technologies available to obtain cost reduction and control of the system. The functional categories in Figure 6 are broken down as follows:

Management Planning - the whole spectrum of management and support (finance, personnel, etc.) not directly related to product design or manufacturing.

Product Engineering - the activities related to product concept, design,testing, and configuration management.

Manufacturing Engineering - those activities associated with the process planning, tool design and planning, N/C programming, industrial engineering, etc. Also included are manufacturing technology, industrial modernization, and facilities.

Factory Operations - the tasks directly related to the manufacture of products, ie, machining, forming, assembly, and all support functions - quality, production control, receiving, shipping, etc.

As shown in Figure 6, information and decision technologies have the broadest potential application across all functions, from the office to the shop floor. While much attention has been given to the mechanization of shop floor processes, the unique characteristics of aerospace (Fig. 3) are such that the integration of data across all functional operations is more fertile ground for significant productivity gains.

CASE STUDIES

Flexible Machining Cell

In July, 1984, VAPD implemented the Flexible Machining Cell (FMC I) as shown in Figure 7. The cell consists of 8 4-axis machining centers, a wash station, 2 coordinate measuring machines, 2 pallet staging carousels, and 4 load/unload stations, all under computer control. The cell is unique, first in that it handles 550 different parts, and secondly in that the factory host communicates all information necessary for the cell to operate without human entry of work orders or any other data. The FMC software architecture is represented in Figure 8. The master machining schedule is created by VAPD's business host computer. The FMC control computer, a DEC 11/44, receives a 20-day window of work orders on a daily basis from the business host. The downloaded work orders are assessed, selected, and scheduled into the FMC by the cell computer control system.

Figure 6. Automation Technologies.

Technology Category	Managemnt Planning	Product Engr.	Manufact. Engr.	Factory Opns.
Shop Floor Equipment				
* Process Equipment				x
* Equipment Controls				x
* Inspection				x
* Material Handling				x
Product Definition				
* Geometric Modeling		x		
* Automated Analysis		x		
* Auto Process Planning			x	
* Automated NC			x	
Information Systems				
* Computer Hardware	x	x	x	x
* Distrib DBMS	x	x	x	x
* Mass Data Storage	x	x	x	x
* Communications/Netwk	x	x	x	x
* Sensors				x
* Part Identification				x
Decision Technologies				
* MRP/Scheduling	x		x	x
* AI/ES/DSS	x	x	x	x
* Simulation		x	x	x
* Group Technology		x	x	

Figure 7. Flexible Machining Cell.

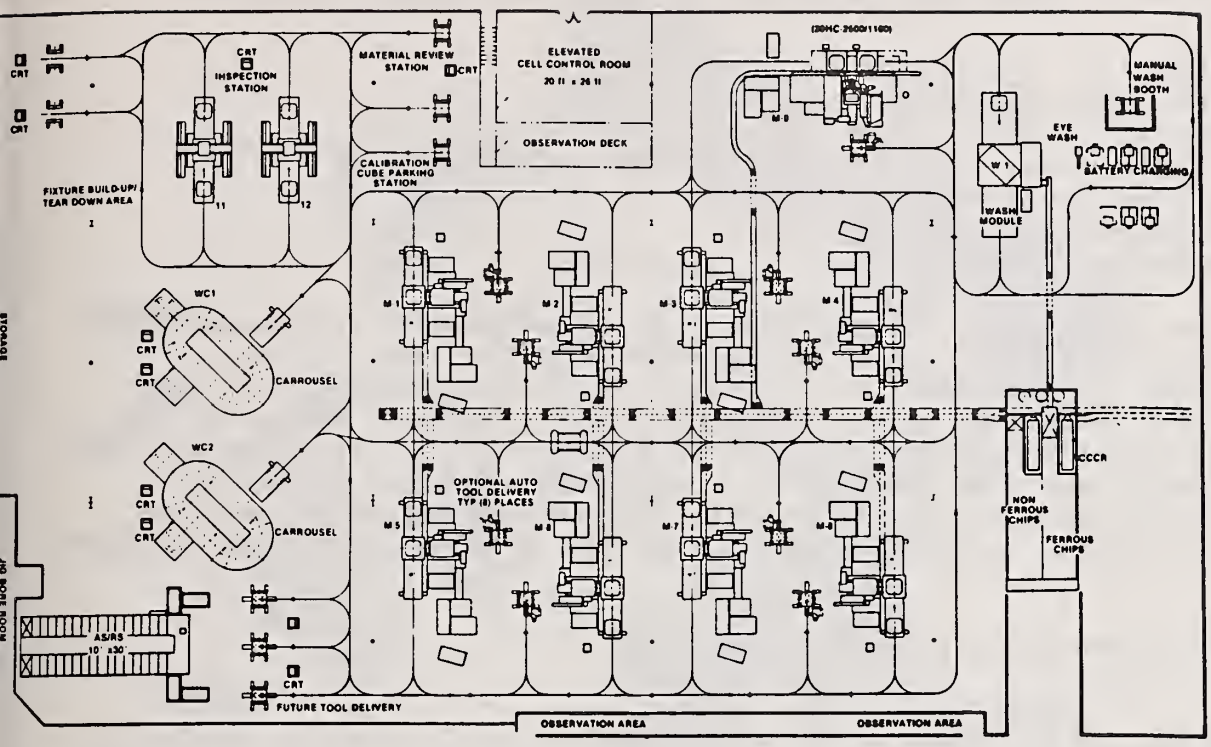
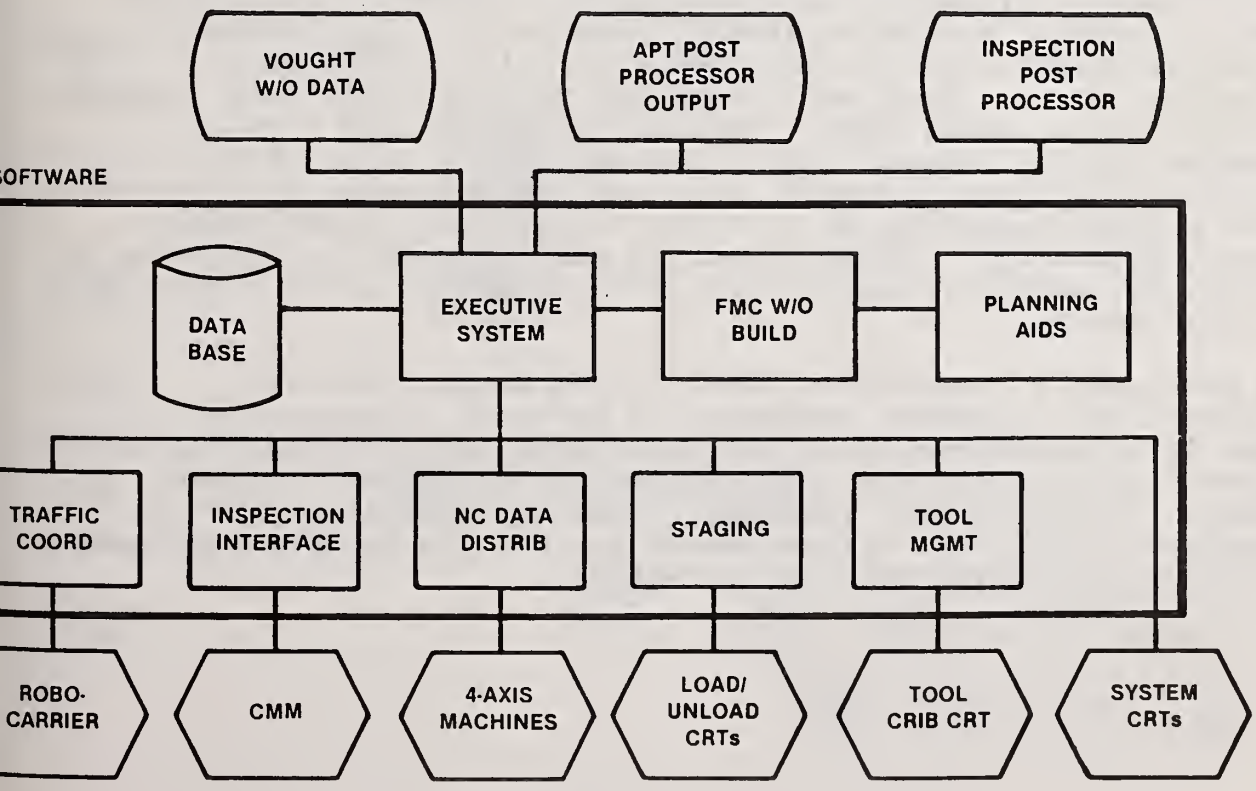


Figure 8. FMC Computer Control Architecture.



When the data is downloaded to the FMC computer, the workload is automatically assessed to maximize machining center resource utilization. This optimization includes consideration of:

- schedule
- work order priority
- material availability
- NC machining and inspection program availability
- cutting tool requirements
- fixtures and pallets
- machining time.

From the downloaded work orders, the FMC computer system selects enough work orders for a 24-hour schedule for the cell's operation. The scheduler optimizes the 24-hour period to meet production schedule, minimize cutter tool changes, maximize machine utilization and to reserve a selection of parts with high machining times for execution on an unmanned third shift. Simulation of selected work loads assures maximized throughput prior to actual authorization during the following 24-hour period.

Integrated Machining System (IMS)

Having gained an experience base with the cell level of control with FMC I, expansion of the machining capability required control at the next level of control, the system. The IMS is VAPD's first project at the system level of control. This system, scheduled for implementation in 1988, is represented in Figure 9. The IMS will integrate prismatic machining cells through a hierarchical computer control system as shown in Figure 10. The prismatic machining capabilities are grouped into cells in the group technology sense. Each cell comprises machining stations, support stations (i.e., wash, CMMs), material handling, and a cell computer control system. FMC I handles 4-axis parts of size envelope 32"x32"x36". FMC II will perform machining for 5 axis parts having dimensions up to 8'x14' using high volume metal removal technology. Other prismatic machining cells, including an FMC III for smaller 5-axis parts, will be added to the system in the near future.

The challenge at the system level is the integration of different cell computer hardware and software architectures. There is also the problem of physical material handling interfaces between cells of different size parts and different equipment vendors. To the extent possible, these considerations must be addressed in the cell design phases so that the integration tasks at the systems level will be minimized.

Figure 9. Integrated Machining System Concept.

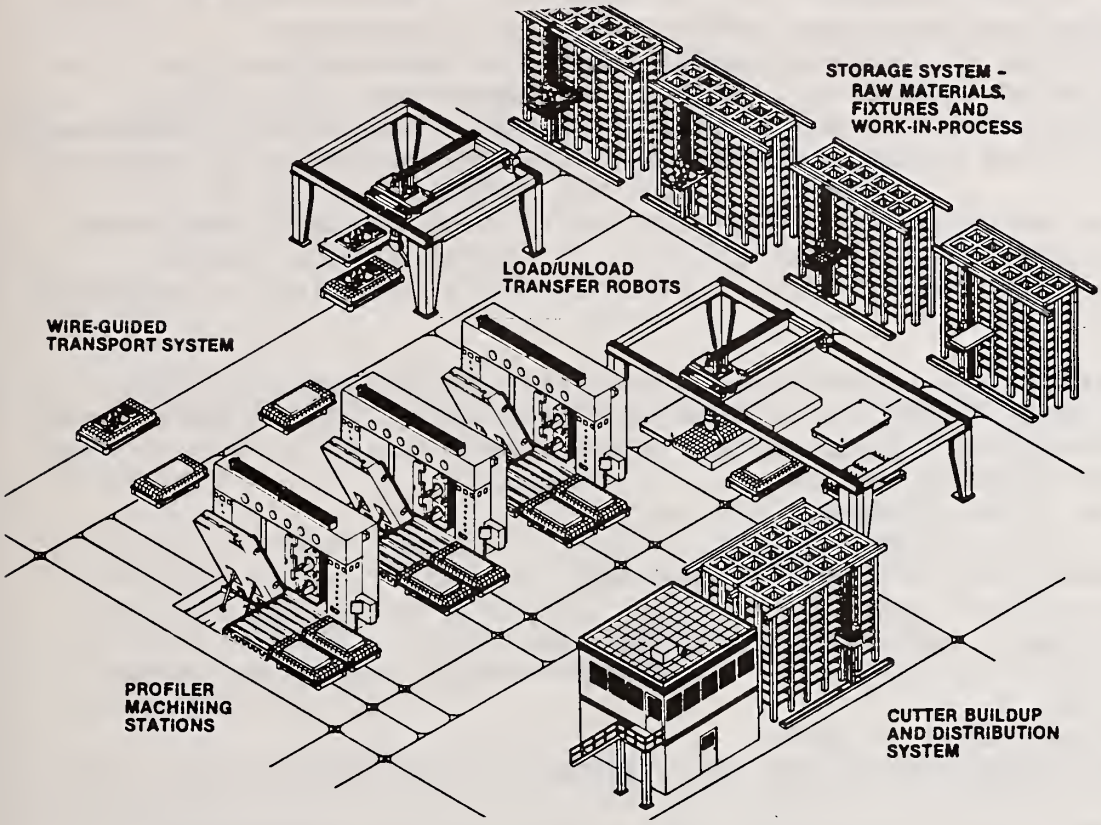
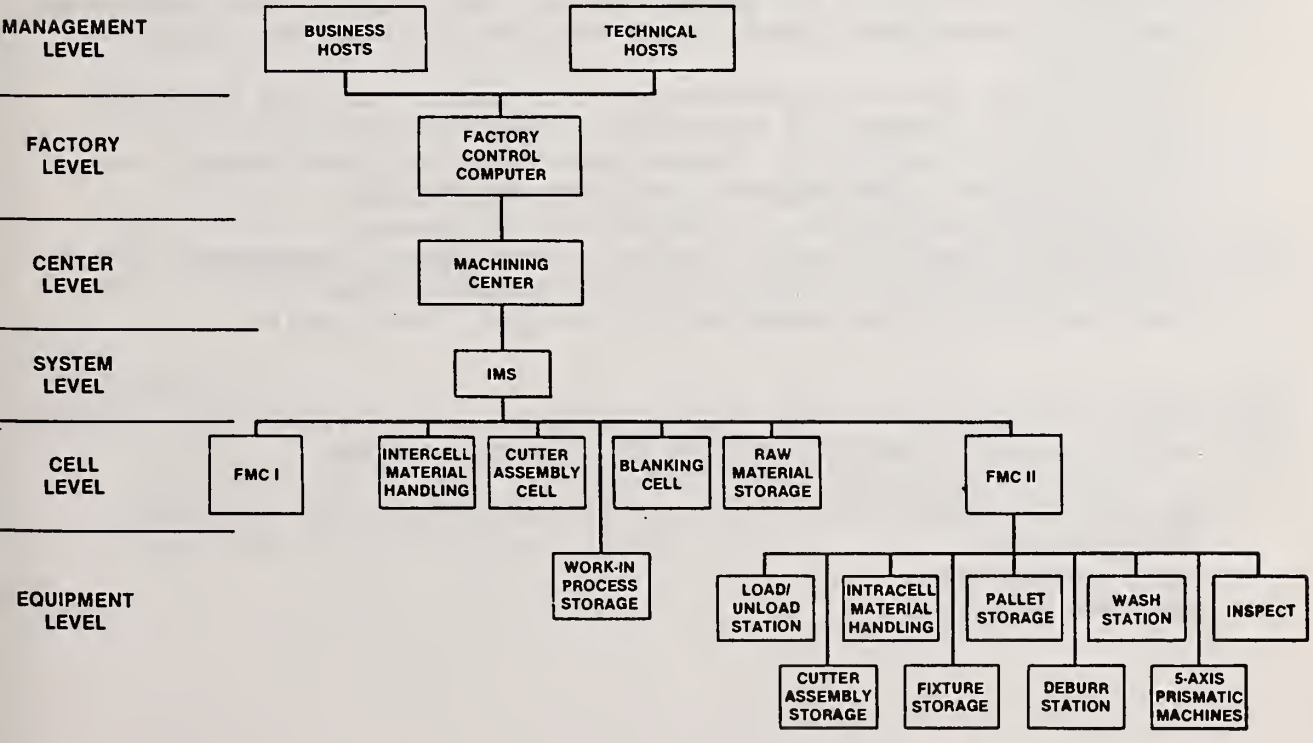


Figure 10. IMS Control Architecture.



Flexible Composites Center

Projections indicate that within the next 5-10 years, aerostructures will contain 50% composites materials. In recognition of this trend, VAPD is implementing a Flexible Composites Center for completion in 1988. (Figure 11). This will be VAPS's first project directed at the center level of control.

Composites manufacturing presents additional control challenges over and above traditional metal parts batch manufacturing processes. The basic flow of composites fabrication is as follows:

1. Receiving, inspection, and storage (cold) of materials.
2. Retrieval, thawing, and movement of materials to layup area.
3. Retrieval and preparation of bond mold tool.
4. Layup of material on mold to form skin.
5. Layup of stiffeners, if any
   - core, foam, channels, etc.
6. Assembly of stiffeners with skin.
7. Bag and pull vacuum.
8. Cure part, generally in autoclave.
9. Transfer cured part to handling fixture, layup tools to storage.
10. Machine edges of part.
11. Dimensional and ultrasonic inspection of part.

Some of the basic automation and control issues related to composites fabrication are:

- evolutionary nature of parts design, materials, and processes,
- identification and tracking of material pieces in the part buildups,
- tracking of time out-of-freezer for each piece of material,
- ad hoc cold storage of wip and unused raw material,
- diversity of part/tool shapes and sizes for handling,
- difficulty of handling uncured part buildups,
- dimensional variations in material thickness,
- scheduling layup of each piece of the layup assembly,
- scheduling of part batches for autoclave cure,
- tool mass/location dependent autoclave cure cycles.

In order to accomplish the varied control tasks within the center, the system level consists of the following modules:

- material control
- fabrication
- post autoclave
- product assurance.

Each of the four systems in the center has a very different set of requirements for control. The material control system is concerned with receipt, inspection, and storage (both cold and ambient) of raw incoming material. The fabrication system controls the time critical scheduling of assembly processes for the layups. The post autoclave system treats the cured composite layups as single part numbers and controls machining operations. The product assurance system manages the inspection activies as well the quality documentation requirements.

SUMMARY

Aerospace manufacturing operations, due to their size and unique requirements, create special challenges for control. There has been some progress in automation and control, examples being the following:

- computer aided design,
- robotics in drilling, welding, routing, deburring, fastening,
- flexible machining systems,
- composite tape laying, rapid ply cutting,
- inspection mechanization and sensors,
- electronic assembly,

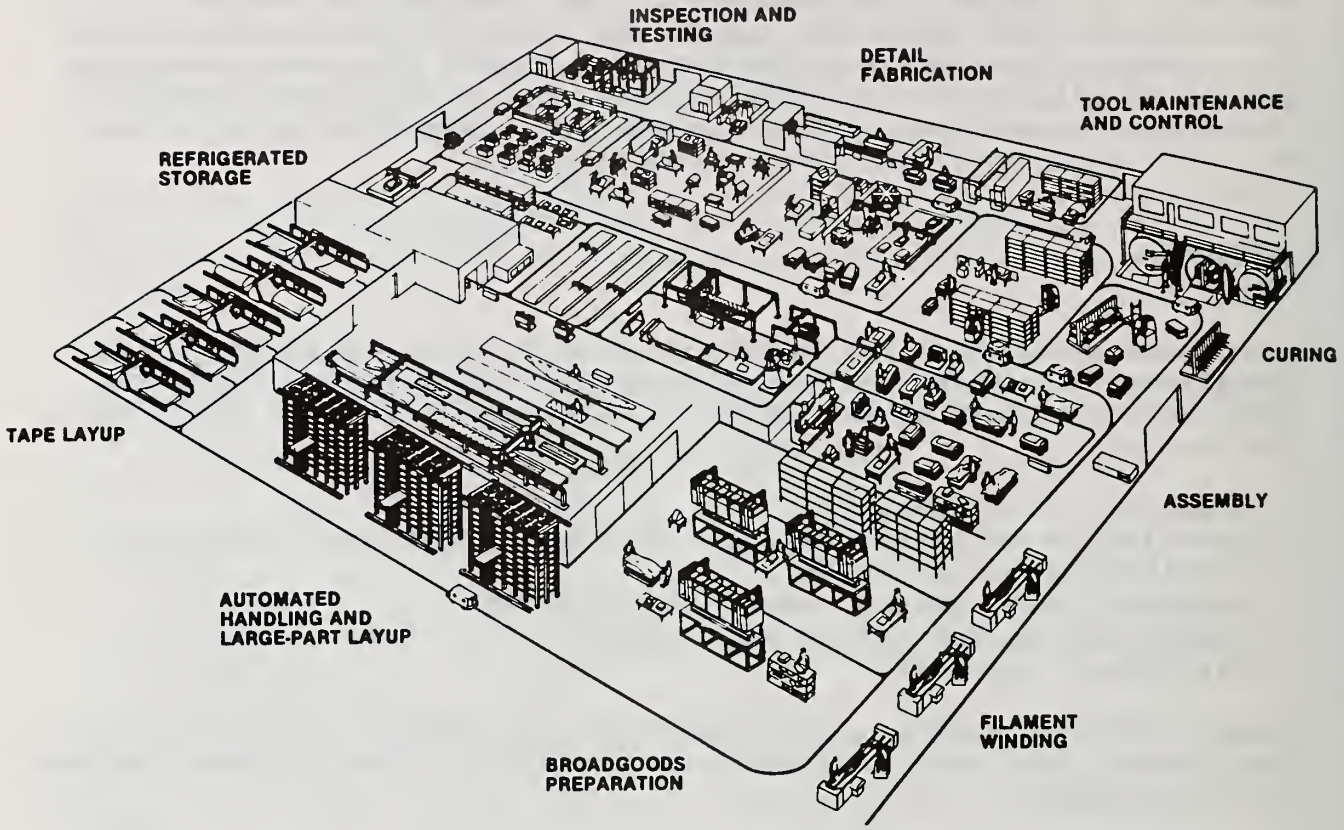
There remains, however, considerable costs and inefficencies in the control and support functions. Specific areas of need include the following:

- finite capacity scheduling of the shop floor,
- scheduling of non-machine and material resources - tools etc.,
- integration of data across functional department boundaries,
- real time data collection,
- solid, feature based geometric models,
- automated generation of NC programs for material nesting, machining, routing, inspection,
- data communications across multi-vendor shop equipment environment,
- management of functional activities to global objectives rather than local performance criteria.

REFERENCES

1. Harkrider, F.E. and S.D. Cook, "Next Level of Control", Flexible Manufacturing Systems '86 Conference Proceedings, CASA/SME, March 1986, Chicago, Ill.

Figure 11. Flexible Composites Center Concept.



REAL-TIME SCHEDULING

OF AN

AUTOMATED MANUFACTURING CENTER[1]

Ram V. Rachamadugu, Narayan Raman, F. Brian Talbot

Graduate School of Business Administration
The University of Michigan
Ann Arbor, Michigan 48109

1.0 Introduction

This paper is part of an ongoing project [see Raman (1985a) and Raman (1985b)] to develop a real-time scheduling system for the Automated Manufacturing Research Facility (AMRF) at the National Bureau of Standards. This facility has been established to serve as a realistic test environment for standards metrology research. [The reader is referred to Jones and McLean (1986) and Simpson, Hocken and Albus (1982) for detailed descriptions of the AMRF.] To reach this goal, computerized planning and control systems are being developed to operate the facility; one component of which is real-time scheduling.

Real-time scheduling in this paper refers to the decision process that specifies which job should be processed next by each machine in the facility, given detailed information about job characteristics (processing time, setup requirements, due-dates, etc.), the current status of the system (e.g., jobs in process, machine status), and scheduling criteria. The present configuration of the AMRF facilitates the decomposition of the problem of scheduling the entire system into single-machine problems for the individual workstations. Consequently, a two-step research approach has been followed. The first step is to examine each machine individually, and to apply and extend, as much as possible, known results from single machine scheduling. The second step is to modify and evaluate the best of these methodologies in an integrated model of the system.

This paper reports on the results of Step One for the Automatic Turning Station (ATS). The ATS manufactures cylindrical parts of several different part types. The workpieces of a given part type are processed in a *batch*; the size of the batch is determined by the geometry of the workpiece and is therefore fixed for a given part type. While changing over from one part type to another, a setup time associated with the change of collets (which hold the workpiece on the machine tool) is incurred. This setup time is, however, independent of the part types involved in the changeover. Customer orders arrive randomly at the ATS; the interarrival time between orders follows no predefined distribution. Each order consists of one or more *jobs* of possibly different part types. Because of the batching requirements, a job of a given part type is considered schedulable only if the number of jobs of that part type is large enough to constitute one or more batches.

---

[1]*The authors thank Drs. Richard Jackson and Albert Jones of the National Bureau of Standards for their valuable assistance in this research effort.*

Though the scheduling problem is essentially dynamic, the lack of a well-defined job arrival process precludes the possibility of building in an intelligent look-ahead procedure which incorporates future job arrivals. This is a major restriction since the knowledge of the distribution of future job arrivals would assist in the formation and sequencing of batches. However, for a truly *random* flexible manufacturing system (FMS) [see Groover (1980)] it is not always possible to characterize the job arrival process through a well-defined distribution. Any attempt to specify such a distribution (based on, say, historical data) presupposes the stationarity of the arrival process. Instead of making such a restrictive assumption, this study treats the dynamic scheduling problem as a series of static problems which are solved on a rolling-horizon basis.

In this paper, we emphasize due-date based scheduling measures which is consistent with the findings of Panwalker, Dudek and Smith's (1973) study of the industrial scheduling problem. Conway, Maxwell and Miller (1967) also note that, for a manager, "the ability to fulfill delivery promises on time undoubtedly dominates other considerations". The scheduling problems investigated are: 1) Minimizing the mean job flow time, 2) Minimizing the average tardiness of all jobs, 3) Minimizing the proportion of jobs tardy, and 4) Minimizing the standard deviation of job tardiness. Though mean job flow time is not a due-date related scheduling measure *per se*, the production system's ability to quote (and maintain) tighter due-dates is improved by minimizing the average time spent by a job in the system; hence the inclusion of this performance measure.

The paper is organized as follows. Section 2 presents a review of the previous research pertinent to this study. Section 3 discusses some characteristics of the mean flow time and mean tardiness problems for the static case. Section 4 extends the investigation to the dynamic case with a discussion of the dispatching procedures studied and the results of a simulation study. Section 5 presents a summary evaluation of the dispatching procedures and the impact of the utilization level on the performance of the ATS. The notation used in the paper is given in Table 1.

2.0 Literature Review

Considerable amount of prior research on scheduling of jobs in conventional job shops already exists. [It is impractical to cite all references here. Good source materials include Conway, Maxwell and Miller (1967), Baker (1974), Graves (1981) and French (1982).] Though conventional job-shops and random flexible manufacturing systems meet the same customer needs (i.e., jobs are made to order), significant differences exist in the system characteristics. Some of these differences which are likely to impact on the scheduling of jobs in FMSs are the large number of alternate routings, buffer limitations, effects of transportation times, low or non-existent setup times and type of scheduling criteria used. Given these significant differences between traditional job-shops and FMSs, it is not clear how much of the conventional scheduling results will carry over to random FMSs. As a consequence, there is an increasing amount of research examining scheduling in FMSs [see, for example, Stecke and Solberg (1981), Lin and Lee (1984), Dar-El and Sarin (1984), Chang, Sullivan and Bagchi (1984), Akella, Choong and Gershwin (1984), Shanker and Tzen (1985) and Afentakis (1985)]. Though some researchers used due-date information for loading decisions, no prior research exists, as far as we know, which addresses the issue of due-date related criteria such as tardiness and proportion of jobs tardy. Though the single-machine problem has a vast literature, few researchers considered the issues relating to batching of jobs or addressed due-date related performance criteria with sequence-dependent setup times. [See Sen and Gupta (1984) for a survey of procedures to schedule jobs against due-date related criteria.] Picard and Queyranne (1978) showed that the single machine problem with a general cost function

Table 1

Notation

p	Part type index, $p = 1,2,..,P$
π_p	Processing time for a job of part type p
N_p	Batch size of part type p
j	Job within a batch (for a given part type), $j = 1,2,..,N_p$, $\forall$ p
B_p	Number of currently schedulable batches of part type p
b	Batch b (for a given part type), $b = 1,2,..,B_p$, $\forall$ p
	For clearer presentation, wherever necessary, b will be used with subscript p to indicate its part type
T	The length of the scheduling horizon $= \sum_{p=1}^{P} (\pi_p N_p B_p) + S \sum_{p=1}^{P} B_p$
t	Time period, $t = 1,2,..,T$
S	Changeover (setup) time from one part type to another
δ_p	1, if the collet currently on the machine is of part type p 0, otherwise
d_{pbj}	Due date of job j in batch b of part type p
T_{pbt}	Tardiness of batch b of part type p if it is completed at time t, $= \sum_{j=1}^{N_p} \max(0, t - d_{pbj})$
X_{bpt}	1, if batch b of part type p is completed at time t 0, otherwise
Y_{cq}^{bp}	1, if batch b of part type p precedes batch c of part type q in the sequence 0, otherwise

and sequence-dependent setup times can be formulated and solved as a time dependent traveling salesman problem. However, they did not consider the issue of batching. Santos and Magazine (1985) developed formulations for the problem of minimizing mean flow time and mean lateness when jobs are required to be processed in batches. Karmarkar, Kekre and Kekre (1984) addressed the issues relating to waiting times in job shops as lot sizes are varied. These two papers do not, however, address the issue of job tardiness.

3.0 Scheduling Approach

Since the distribution of the job arrival process is not known *a priori*, we have decided to decompose the dynamic problem into a series of static problems to facilitate real-time implementation of any scheduling procedure. A static problem is generated whenever the ATS becomes available, and it involves making a decision on the job to be processed next based only on those jobs which are available in the system at that point in time. In the remainder of Section 3, we present some theoretical results for the static mean flow time and mean tardiness problems. These results are incorporated into the dispatching procedures investigated in Section 4.

The static scheduling problem is characterized by: 1) Formation of batches of jobs of fixed size for a given part type, and 2) A constant changeover time between one part type and another. Note that the requirement that jobs of the same part type be processed in a batch implies that the completion times of all jobs in a batch equal the completion time of the last job in the batch.

At any given point in time, the ATS can be in any one of two states – i) State A, in which there is no collet on the machine, if for instance, the machine was torn down just prior to the period in which the scheduling decisions need to be made, and ii) State B, in which there is a collet for a specific part type on the machine. Because of the changeover time, the optimal decision would in general depend upon on the current state of the machine.

To distinguish this problem from single-machine problems without batching requirements and changeover times, we will henceforth refer to the latter as *regular* single-machine problems.

3.1 Minimizing Mean Flow Time

To determine the sequence optimal for minimizing mean flow time, polynomially bounded procedures can be developed based on the following theorem.

Theorem 1: *Given that the ATS is in state A, the mean flow time of all jobs is minimized by grouping the batches of the same part type together, and sequencing the part types in the non-decreasing order of their weighted batch processing times (WBPT),* $S/N_p + \pi_p$.

Proof: Refer to Rachamadugu, Raman and Talbot (1986).

Note that the condition stated in Theorem 1 is equivalent to the Weighted Shortest Processing Time rule for the regular single-machine problem, in which a batch of jobs is treated as a single job with processing time equal to the sum of the processing times of all jobs in the batch, and with a weight equal to the number of jobs in the batch.

If the ATS is in state B, the WBPT sequence need not be optimal, since sequencing the batches of the part type for which the collet is currently on the machine first may lead to a reduction in the mean flow time. The relative positions of the other batches in the WBPT sequence would, however, remain the same. To illustrate this case, refer to Figure 1.

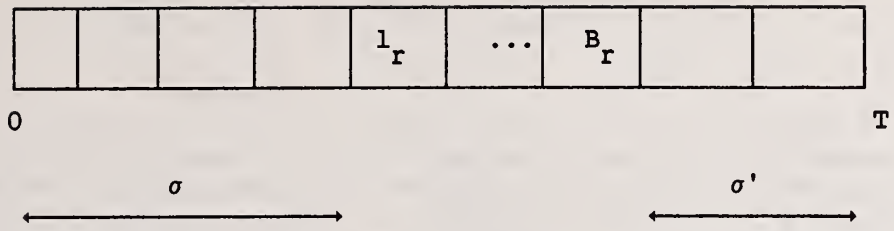


Figure 1 – The Gantt Chart for WBPT Sequence

Let r be the part type for which the collet is currently on the ATS. Let σ (σ') be the set of jobs which precede (succeed) jobs of part type r. Then, by sequencing part type r first, the increase in the flow times of jobs in σ,

$$X = B_r \pi_r N_r \sum_{i \in \sigma} B_i N_i$$

The decrease in the flow times of jobs of part type r,

$$Y = B_r N_r \sum_{i \in \sigma} (B_i \pi_i N_i + S)$$

The decrease in the flow times of jobs in σ',

$$Z = S \sum_{j \in \sigma'} B_j N_j$$

The WBPT sequence is optimal if $X \geq Y+Z$, otherwise the sequence in which the batches of part type r are scheduled first is optimal.

3.2 Minimizing Mean Tardiness

The complexity of the mean tardiness problem even for the regular single-machine case has long remained an open question. The following theorems, however, establish some characteristics of the optimal sequence for the problem under study.

Theorem 2: *In a sequence optimal for the problem of minimizing mean tardiness, the batches of jobs for a given part type are sequenced in the non-decreasing order of the due dates of the jobs.*

Proof: Refer to Rachamadugu et al. (1986).

It follows from Theorem 2 that, while seeking the optimal solution, the waiting jobs of a given part type p can be ordered in the Earliest Due Date (EDD) sequence, and batches can be formed from this ordered list by grouping the first N_p jobs, then the next N_p jobs, and so on. The jobs remaining after the last batch has been formed need to be considered only in the next cycle, i.e., when the ATS next completes a batch of jobs. The implied precedence relationship among batches yielded by this procedure is depicted in Figure 2.

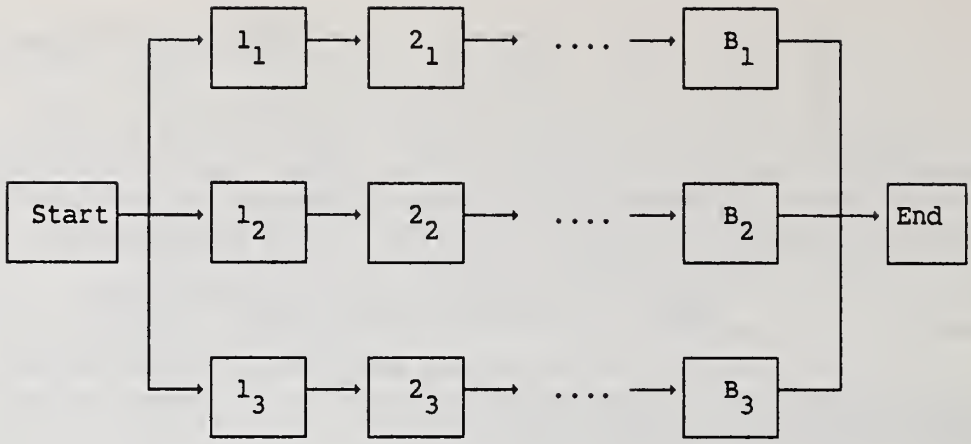


Figure 2 – The Precedence Relationship among Batches in an Optimal Sequence

Henceforth, unless stated otherwise, it is assumed that jobs of the same part type are batched in the manner stated above.

Theorem 3: *Given that there is no more than one batch of each part type, any two adjacent batches, x of part type p, and y of part type q, satisfy the following condition in an optimal sequence,*

$$a_y C_x - a_x C_y + \sum_{j \in \Delta_y} (K - d_{qyj}) - \sum_{i \in \Delta_x} (K - d_{pxi}) \leq 0$$

where,

a_x (a_x)	Number of jobs which are tardy in x (y) when it precedes y (x)
C_x (C_y)	$S + \pi_p$ ($S + \pi_q$)
K	Completion time of the batch scheduled in the later position
Δ_x (Δ_y)	Set of jobs which are not tardy when x_p (y_q) precedes y_q (x_p) but which are tardy when the positions of these batches are interchanged

Proof: Refer to Rachamadugu et al. (1986).

Theorem 3 states the necessary condition for optimality for the case when there is no more than one batch of each part type. It can be easily verified that this condition reduces to the modified due-date rule (refer Baker and Kanet (1983)) which is a necessary condition for the regular single-machine problem (Rachamadugu (1985)). As shown in Rachamadugu et al. (1986), the condition stated in Theorem 3 is not necessary for the general case. It can nevertheless be used to drive a heuristic procedure and as long as the changeover time S is small relative to the batch processing times $\pi_p N_p$, the violation of the basic assumption should have only a minor impact on the solution quality. Note that the condition stated in Theorem 3 can establish precedence between batches x and y only when at least one of these two batches contains one or more late jobs when it is scheduled in the later position. Also, the precedence relationship established by this theorem is not necessarily transitive.

A dynamic programming formulation of the mean tardiness problem is given in Rachamadugu et al. (1986). Although conceptually interesting, the dynamic programming approach suffers from the 'curse of dimensionality', and from not providing a feasible solution prior to complete generation of the state space. Both drawbacks can be ameliorated by incorporating the precedence relationships depicted in Figure 2, along the lines suggested by Baker and Schrage (1978), or by incorporating heuristic bounds. However, the experience of Wee and Magazine (1981) and Talbot, Patterson and Gehrlein (1986) on related scheduling problems indicated that we would be better off pursuing the integer programming approach discussed below. The depth-first implicit enumeration procedure developed has the benefits of requiring little memory (and unlike dynamic programming, the amount of memory required is known before the problem is solved), and it can be stopped prior to optimality to yield a good feasible solution.

Integer Programming Formulation

An integer programming formulation for the tardiness problem is shown below:

$$\text{Minimize} \quad \sum_{p=1}^{P} \sum_{b=1}^{B_p} \sum_{t=1}^{T} T_{bpt} X_{bpt}$$

Subject to,

$$\sum_{t=1}^{T} t X_{cqt} - \sum_{t=1}^{T} t X_{bpt} + M (1 - Y_{cq}^{bp}) - N_q \pi_q - S' \geq 0 \qquad (1)$$

$$1 \leq t \leq T: \; p,q \in \Pi: \; 1 \leq b \leq B_p: \; 1 \leq c \leq B_q$$

$$\sum_{t=1}^{T} t X_{bpt} - \sum_{t=1}^{T} t X_{cqt} + M Y_{cq}^{bp} - N_p \pi_p - S' \geq 0 \qquad (2)$$

$$1 \leq t \leq T: \; p,q \in \Pi: \; 1 \leq b \leq B_p: \; 1 \leq c \leq B_q$$

$$X_{bpt}, \; X_{cqt}, \; Y_{cq}^{bp} \in \{0,1\}, \quad \forall \; p,q,b,c,t \qquad (3)$$

where,

M = A large positive number, and

$$S' = \begin{cases} S, \text{ if } p \neq q \\ 0, \text{ otherwise} \end{cases}$$

Constraints (1) and (2) represent the disjunctive relationships between batches in the precedence network depicted in Figure 2.

This formulation leads to a large number of constraints and variables even for moderate-sized problems. The proposed implicit enumeration solution approach, however, obviates the need for generating or testing constraints (1) and (2) explicitly, and efficiently exploits the characteristics of the structure depicted in Figure 2. The solution procedure uses depth-first search and builds the schedule forwards in time. A node at level L in the enumeration tree corresponds to a partial sequence of L batches. Any given node n in the tree has an associated array A_n which contains the indexes of batches which are schedulable at the next level. The precedence relationships lead to a significant reduction in the computer storage requirements since the cardinality of A_n is limited to the total number of the part types of the unscheduled batches.

Starting from the unique node at level 0, the procedure implements the Modified Myopic rule (to be discussed in Section 4.1) at each node to determine the relative priorities of batches potentially schedulable at the next level. These batches are maintained in the non-increasing order of their priorities in array A_n. Augmentation at node n at level L requires selecting the batch with the highest priority (as determined a priori by the Modified Myopic rule) among those which have not already been branched from.

The lower bound at node n is given by,

$$LB(n) = T(\sigma) + \max(0, L(\sigma'))$$

where $T(\sigma)$ is the tardiness already incurred by batches scheduled according to sequence σ, and $L(\sigma')$ is the total lateness incurred by scheduling the remaining batches in the WBPT sequence. Since the WBPT sequence minimizes mean flow time, it minimizes mean lateness as well, and also since lateness is an underestimate of tardiness for any feasible sequence, this bound is valid.

Since the breadth of the tree is determined by the cardinality of A_n associated with each node n at the preceding level, computational requirements increase exponentially with an increase in the number of schedulable part types. Fortunately, however, unless all jobs have substantial slack, the effectiveness of the lower bounding procedure, which is dependent upon the number of batches with late jobs, generally improves with an increase in the number of part types. In general, this procedure is more useful at the middle and the lower parts of the tree. At very low levels in the tree, however, the computational overhead associated with the repetitive calculation of LB(n) precludes its usage.

The results of the computational studies with this enumeration procedure for the dynamic problem are discussed in Section 4.

4.0 Dynamic Scheduling

As mentioned in Section 1.0, the dynamic nature of the real-time scheduling problem requires implementation of dispatching procedures for selecting the batch to be processed next whenever the ATS becomes available. We describe the dispatching rules investigated in Section 4.1. Section 4.2 describes the simulation experiment conducted to evaluate the effectiveness of these dispatching rules. The results of the experiment are discussed in Section 4.3.

4.1 Description of the Dispatching Procedures

This paper investigates nine dispatching procedures. The first procedure is based on the first-come-first-serve discipline. The next four procedures – the Revised Modified Due Date rule, the Modified Myopic rule, the Modified Montagne's rule and the Revised Earliest Due Date rule are modifications of heuristic methods found to be effective for the regular single-machine and/or job-shop problems by various researchers. The modifications have been necessitated by the need to incorporate the batching of jobs and changeover times. The sixth procedure is based on the Weighted Batch Processing condition discussed in Section 3.1. The seventh procedure is derived from the necessary condition for local optimality stated in Theorem 3, while the eighth procedure is a combination of the necessary condition and the Modified Myopic rule. The final procedure tested is based on the implicit enumeration approach described in Section 3.2. These procedures are now discussed.

(i) First Come First Serve (FCFS) Method

The First Come First Serve rule sequences the batches in the order in which they are formed. This rule essentially provides a benchmark for establishing the relative effectiveness of the other procedures investigated.

(ii) Revised Modified Due Date (RMDD) Method

In their studies of the single-machine and job-shop tardiness problems, Baker and Bertrand (1982), Baker and Kanet (1983) and Baker (1984) found the Modified Due date (MDD) rule to be quite effective relative to other heuristics under varying levels of machine utilization and due-date tightness, and for different due-date setting rules. The robustness of the MDD rule lies in effectively combining the Shortest Processing Time (SPT) rule, which has been shown to be quite effective when the due dates are set very tightly, and the EDD rule which performs well when the due dates are lax. Rachamadugu (1985) has shown that, for the regular single-machine tardiness problem, the MDD rule is a necessary condition for optimality.

For the present study, the MDD rule has been modified as follows. The revised modified due date $RMDD_{bp}$ of a batch b of part type p is defined as,

$$RMDD_{bp} = \sum_{j=1}^{N_p} \max (t + \delta_p S + \pi_p N_p, d_{pbj})$$

where t is the time at which the scheduling decision has to be made.

According to the RMDD rule, when the ATS becomes available at time t, the batch with the earliest modified due date $RMDD_{bp}$ is selected for processing. While it can be shown that the RMDD rule is not necessary for optimality, it is nevertheless worth investigating, given the strength of the MDD rule for regular single-machine problems.

(iii) Modified Myopic (MYOP) Method

Morton and Rachamadugu's (1982) study of the single-machine weighted tardiness problem found the Myopic rule to be quite effective compared to the other rules tested. The Myopic rule is similar to the MDD rule in the sense that it reduces to the SPT rule when all jobs have non-positive slack, and to the EDD rule when they have substantial slack. In the intermediate range, however, the priority assigned to a job increases exponentially with decrease in its slack. The Modified Myopic rule assigns priority $MYOP_{bp}$ to a batch b of part type p, where,

$$MYOP_{bp} = \sum_{j=1}^{N_p} Pr_{bpj} \text{ and,}$$

$$Pr_{bpj} = 1/(\delta_p S + \pi_p N_p)\, e^{[-(d_{pbj} - (t + \delta_p S + \pi_p N_p))^+ / \pi_{ave}]}$$

where, π_{ave} is the mean processing time of a batch of jobs, averaged over all part types, and $(x)^+$ denotes max (x, 0).

(iv) Modified Montagne's (MONT) Method

In their experimental investigation of the regular single-machine tardiness problem, Baker and Martin (1974) found the dispatching procedure suggested by Montagne (1969) to perform well. In our study, this procedure has been revised to yield the Modified Montagne's rule which sequences the available batches in the non-decreasing order of $MONT_{bp}$, where for batch b of part type p,

$$MONT_{bp} = \sum_{j=1}^{N_p} [(\pi_p N_p)/(\sum_{p=1}^{P} \sum_{b=1}^{B_p} (t + \pi_p N_p + S) - d_{pbj})]$$

The implementation of the RMDD, MYOP and MONT rules, automatically incorporates the precedence relationship shown in Figure 2. Computational requirements are thereby reduced by restricting the evaluation of priorities, for each part type, to only the batch with the lowest sum of job due dates.

(v) Revised Earliest Due-Date (REDD) Method

The REDD method sequences the batches in the non-decreasing order of the sum of the due-dates of jobs in the batch. Operationally, this rule is equivalent to the Earliest Due Date rule for regular single-machine problem.

(vi) Weighted Batch Processing Time (WBPT) Method

The WBPT method sequences the batches in the non-decreasing order of their weighted batch processing times $S/N_p + \pi_p$. Ties between batches are broken by sequencing the batch with the lower sum of job due-dates first. As mentioned in Section 3.1, this method is equivalent to the Weighted Shortest Processing Time rule for regular single-machine problems.

(vii) Necessary Condition-Based (NC) Method

This method implements the condition stated in Theorem 3 as a heuristic procedure. Though this theorem is restrictive in its scope, it is obvious from the discussions presented in Section 3.2 that the application of this theorem to the more general case, in which there is more than one batch of each part type, leads to minor degradation of the solution quality, as long as the changeover time S is small relative to the batch processing times $\pi_p N_p$.

The NC method considers the schedulable batches two at a time and establishes the precedence relationship between them. As discussed in Section 3.2, Theorem 3 may not always be able to establish the precedence relationship between two batches. In such cases, the NC method determines the precedence in the favor of the batch with the lower sum of job due-dates. In general, since this precedence relationship need not be transitive, cycles may be formed while applying this procedure. In this study, however, the formation of such cycles is avoided by enforcing transitivity in the order in which the batches are considered.

(viii) Revised Necessary Condition-Based (RNC) Method

This procedure is similar to the NC method, the underlying difference being the use of the Modified Myopic rule, instead of the sum of the job due dates, to break ties between two batches when Theorem 3 fails to establish precedence.

The precedence relationships imply that, while implementing the NC and the RNC procedures, the evaluation of the condition stated in Theorem 3 can be restricted to the batches due the earliest for each part type.

(ix) Implicit Enumeration (BB) Method

This procedure implements the branch and bound method discussed in Section 3.2 to solve the static tardiness problem optimally for selecting the batch to be processed next. While this method does not guarantee optimality for the dynamic-scheduling problem, its effectiveness in a rolling-horizon environment seemed worth investigating. Unlike the procedures mentioned earlier, this method provides an ordered sequence of all batches. Clearly, if no batch has been formed since the previous implementation of this procedure (in the previous cycle), the same sequence can be retained for this cycle as well. Operationally, this reduces the number of times the static optimal solution must be found.

4.2 The Simulation Model

The simulation model considers ten part types. The processing times π_p and the batch sizes N_p are shown in Table 2. Currently, all part types being manufactured at the ATS have a batch size of 6 which has been retained in this study for part types 1 through 8. To permit generalization, part types 9 and 10 with batch size of 3 have been included.

To capture the lack of a well-defined order arrival process in the best manner possible, the order interarrival times are assumed to be exponentially distributed, which provides a large variance in the sampled values. The number of jobs in an order varies uniformly between 1 and 10, each job being equally likely to belong to any one of the ten part types being manufactured currently. The mean job interarrival time determines the ATS utilization. In this study, this arrival rate is varied to yield three levels, 50%, 70% and 90%, of ATS utilization.

Table 2

Part Details ·

Part Type	Batch Size	Processing Time (Minutes)
1	6	1.0
2	6	2.0
3	6	3.0
4	6	4.0
5	6	5.0
6	6	6.0
7	6	7.0
8	6	8.0
9	3	8.0
10	3	9.0

Changeover time between part types : 1.0 minute

Upon its arrival, each job is assigned a due date based on the Total Work Content (TWK) rule. This rule has been found to be quite effective in previous studies (see, for example, Baker (1984)). According to this rule, the due date assigned to a job is the sum of its arrival time, and a flow time which is the product of the job processing time and the flow allowance factor. The flow allowance factor essentially controls the tightness of the due dates. In the present study, this rule was modified to include the average time a job has to wait for batching as well. The due date $d_{p.j}$ assigned to job j of part type p is given by,

$$d_{p.j} = a_j + F \pi_p N_p + \beta_p \qquad (4)$$

where a_j is the arrival time of job j, F is the flow allowance factor and β_p is the average batching time for a job of part type p. For the simulation runs, a range of due-date tightness is achieved by using flow allowance factors of 2, 4, 5, 7, 9, 10, 11, 13, 15, and 20. β_p is determined by the batch size N_p and the average interarrival time Λ_p between two successive jobs of part type p from the following relationship,

$$\beta_p = \Lambda_p (N_p - 1) / 2,$$

and,

$$\Lambda_p = \Lambda / (\theta_p O_{ave})$$

where Λ is the average interarrival time between two successive orders, θ_p is the probability that an arriving job is of part type p, and O_{ave} is the average order size.

Two measures were used to confirm the validity of the sampling process. The realized utilization levels were found to lie within 1.5% of the theoretical values of 50%, 70% and 90%, and the realized batching times were within 5.6% of the theoretical values for the ten part types. Since the flow times and the tardiness values of the jobs within the same batch are highly correlated, these two statistics were recorded as single observations for the entire batch.

The simulation run covered a steady-state period of 14400 minutes. Observations over this period were batched; the correlation between batches was found to be insignificant at the 95% confidence level.

4.3 Simulation Results and Discussion

4.3.1 Experimental Results

The results of the simulation runs are presented below for each of the four performance criteria mentioned above.

Mean Flow Time (MFT)

For all the nine procedures, the mean flow time values exhibit a V-shaped curve when they are plotted against machine utilization levels. The values at 70% utilization are consistently lower than the corresponding values at 50% and 90% utilizations. Figure 3 depicts this behavior for the MYOP rule.

Among the dispatching procedures, WBPT is dominant at all utilization levels and at all flow allowance factors. With the exception of NC and RNC, the other rules yield comparable results at 50% utilization; as the level of utilization increases, however, the best results are yielded by BB and MYOP at low values of F while MONT is clearly superior at very high values of F. While FCFS and WBPT yield constant values of MFT at all values of F for a given utilization, the other rules exhibit varying degrees of sensitivity to due-date tightness, especially so at 90% utilization. NC and RNC, in addition to yielding very high values of MFT, appear to be very sensitive to F.

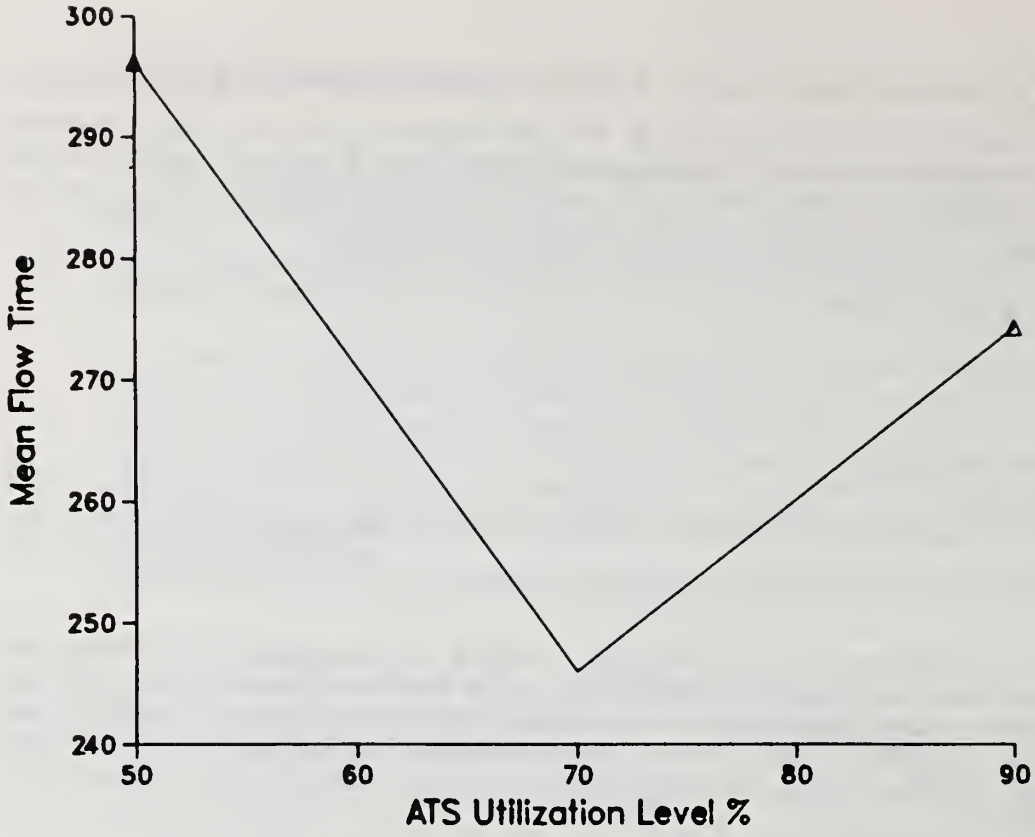


Figure 3 – Graph of Mean Flow Time as a function of ATS utilization level for MYOP.

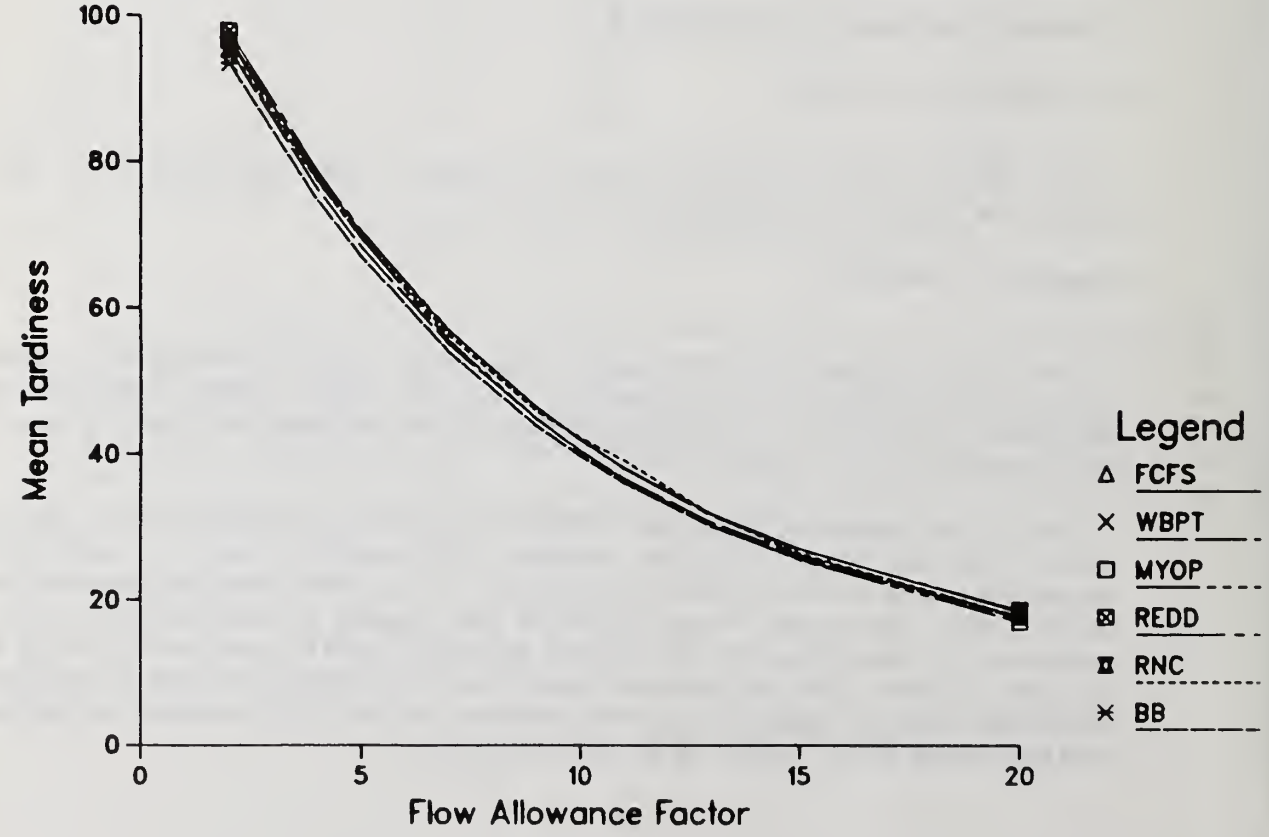


Figure 4 – Graph of Mean Tardiness as a function of Flow Allowance Factor at 50% utilization level.

Mean Tardiness (MT)

For all the procedures investigated, the plot of MT against utilization level exhibits the V-shape for low values of F. However, the curve tends to become flat as F increases, in particular the relative increase in MT from 70% utilization to 90% utilization decreases at increasing rate. For FCFS, WBPT and MONT, however, the values of MT at 90% utilization continue to be higher than the corresponding values at 70% even at very high values of F. For other rules this crossover occurs at different values of F with BB and MYOP crossing over the earliest. For all the rules, the MT values at 50% utilization are high across all values of F.

The values of MT as a function of F are depicted in Figures 4 through 6 for the utilization levels of 50%, 70% and 90% respectively.[2] BB has the best overall performance across all ranges of due-date tightness and utilization levels. However, MYOP, and to a lesser extent RNC, are also very efficient; their relative performance improves with an increase in the utilization level. The relative efficiencies of these rules are most prominent at low values of F. Though the FCFS rule appears to be effective at 50% utilization, its performance deteriorates rapidly with increase in the level of utilization. With a reduction in due-date tightness, the values of MT obtained under the different rules tend to converge; the convergence is, however, less rapid as the utilization level increases.

Proportion of Jobs Tardy (PT)

The V-shape is less obvious for the measure of PT when it is plotted against the utilization level. At low values of F, PT increases with increase in utilization level, the increase is more rapid in the 70%−90% range. As F increases, the curve tends to become flatter, the V-shape appearing for the intermediate values of F. As F increases, PT decreases monotonically with increase in utilization levels.

At 50% utilization, BB, MYOP and WBPT yield the best values; BB and MYOP do so for high flow allowances at 70% and 90% utilization levels as well. However, for tight due dates MONT is distinctly the best at 70% utilization while WBPT is dominant is 90% utilization. Also, as the utilization level increases, the performance of FCFS degrades rapidly while NC and RNC appear to be increasingly effective. Figure 7 depicts the behavior of PT as a function of F at 90% utilization level.

Standard Deviation of Tardiness (SDT)

The values of SDT against ATS utilization levels depict the familiar V-shape at low values of F, the degradation at lower values of utilization is quite sharp.

There is only a marginal difference among the dispatching rules at 50% utilization. However, as the utilization level increases, BB, MYOP, NC and RNC appear to be superior to the other procedures and are equally effective at high flow allowances, while at 90% utilization and for extremely tight due-dates, FCFS is clearly

---

[2] *For greater clarity in presentation, we have omitted from the graphs the results of those dispatching procedures which have been consistently dominated by others. The results of FCFS, REDD and WBPT have, however, been retained to serve as benchmarks.*

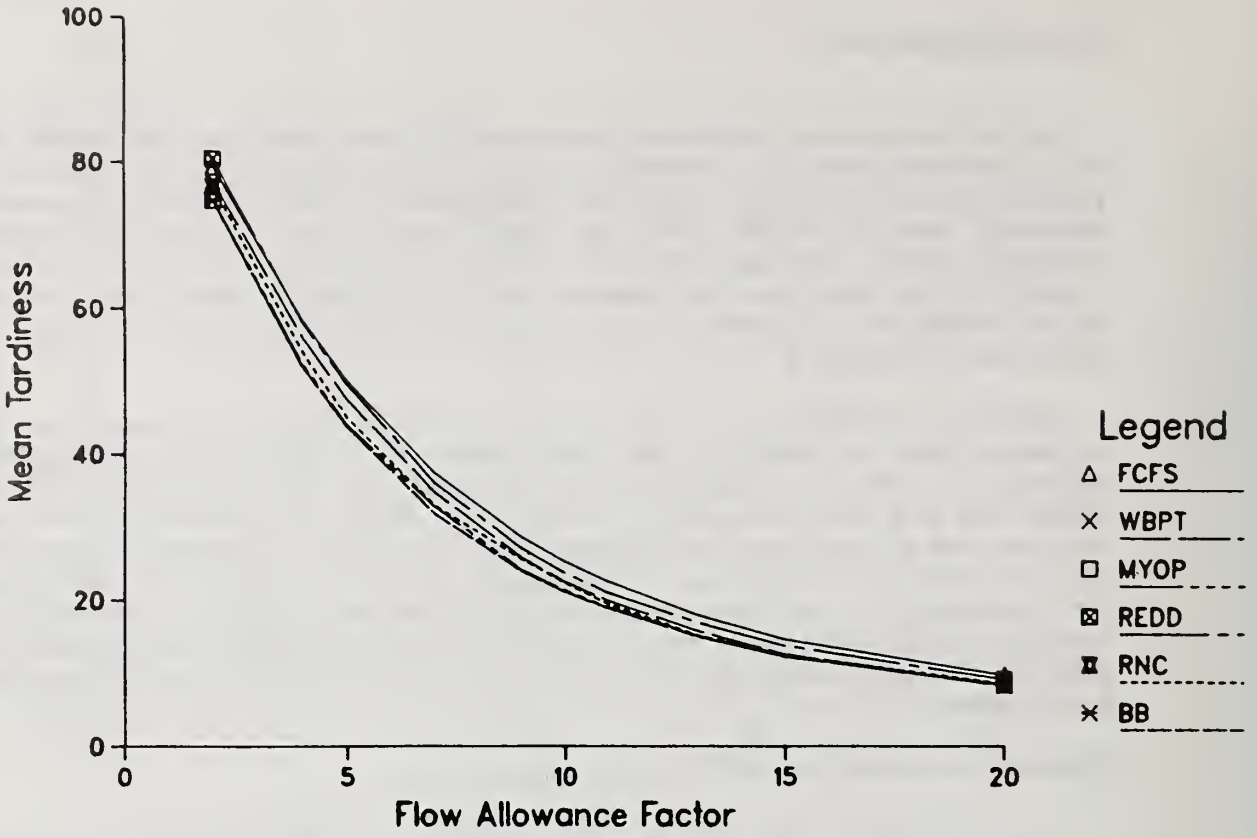


Figure 5 – Graph of Mean Tardiness as a function of Flow Allowance Factor at 70% utilization level.

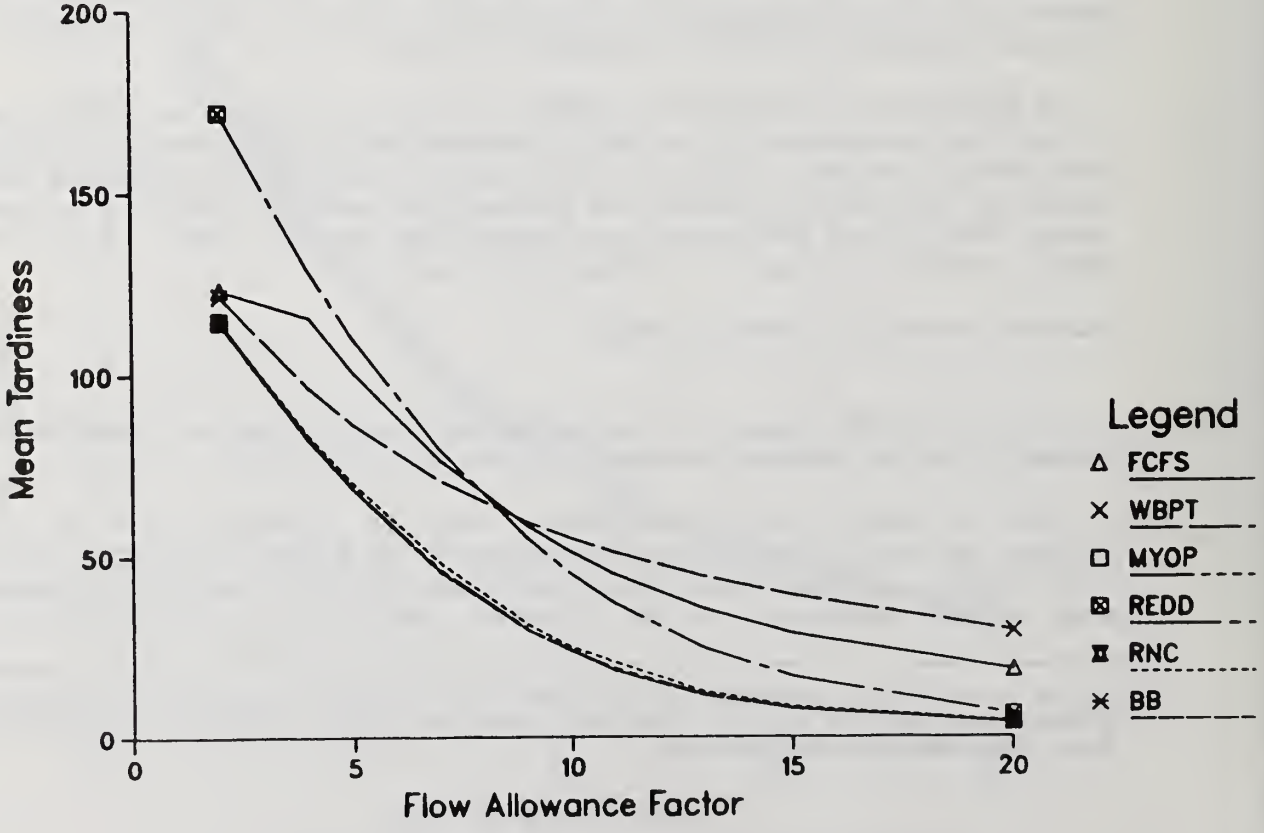


Figure 6 – Graph of Mean Tardiness as a function of Flow Allowance Factor at 90% utilization level.

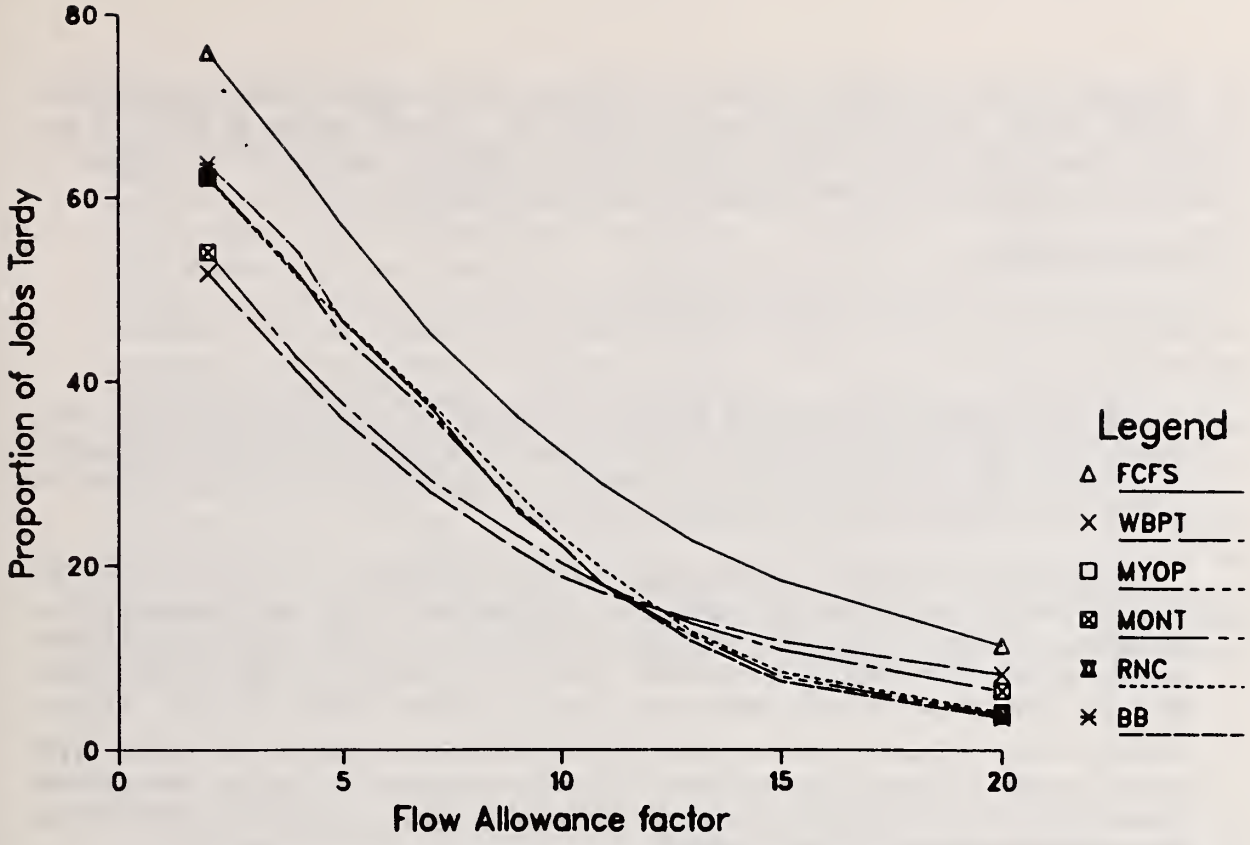


Figure 7 – Graph of Proportion of Jobs Tardy as a function of Flow Allowance Factor at 90% utilization level.

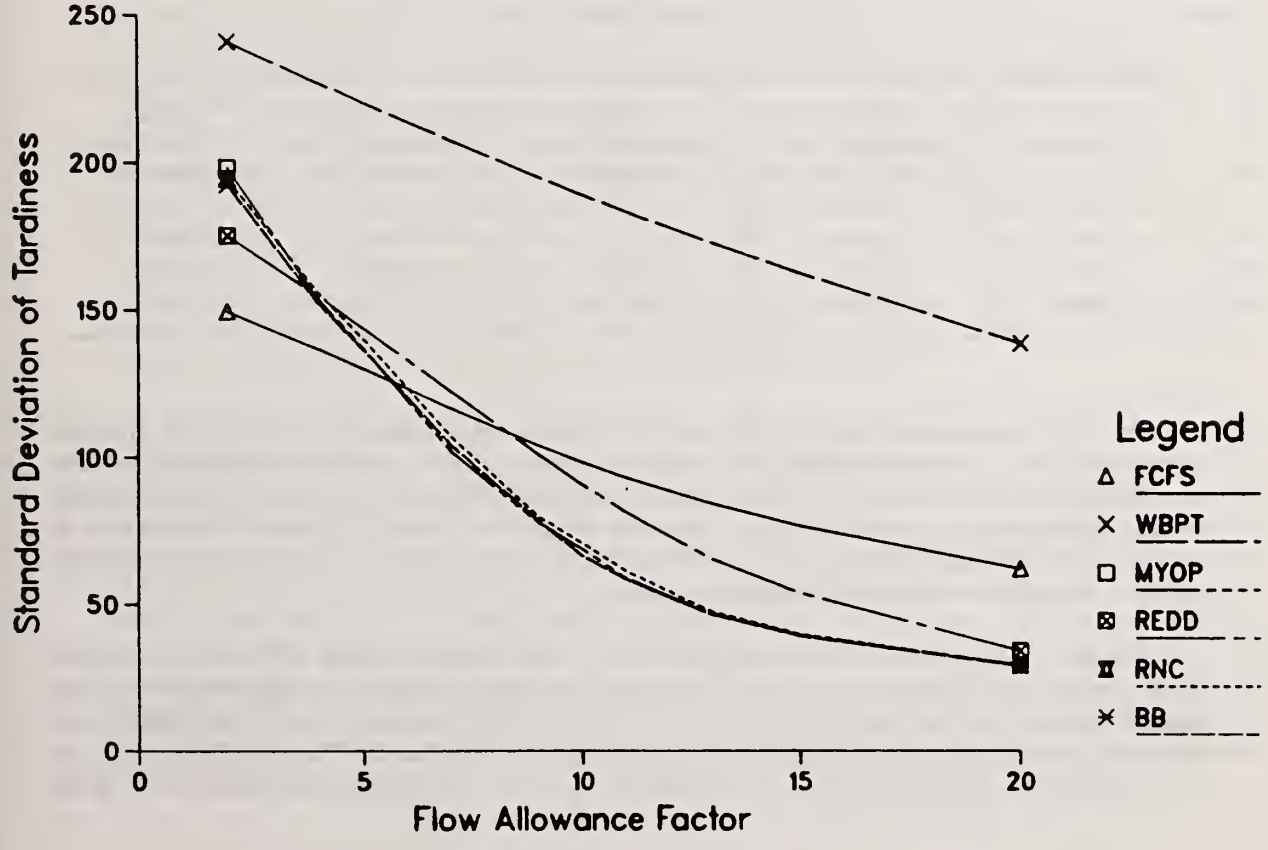


Figure 8 – Graph of Standard Deviation of Tardiness as a function of Flow Allowance Factor at 90% utilization level.

dominant. WBPT and MONT yield the worst results; their relative performances degrade further as the utilization level increases. Also, the crossover values of F for all the dispatching rules are lower than the corresponding values for the PT criterion. Figure 8 compares the behavior of the procedures for the 90% utilization level.

4.3.2 Discussion

Impact of Utilization Level and Flow Allowance Factor

All dispatching procedures yield V-shaped curves for the four measures studied. This result is quite prominent for the MFT criterion where this shape is retained for all values of F. For the MT and the SDT criteria this shape is observed for low values of F while, for the PT criterion, it occurs at intermediate values of F.

Note that the flow time of any job comprises its batching time – the time it waits for the other members of its batch to arrive, its queueing time – the time it spends (after batching) in waiting for the ATS to complete processing of other jobs ahead of it, and the processing time of the batch it is a member of. At 50% utilization, the large values of flow time are attributable to high interarrival times Λ_p which results in large batching times. At 90% utilization, the increased number of batches vying for the use of ATS results in large values of queueing times. A typical example of the relative values of the average batching, the average queueing and the average processing times for FCFS at the three utilization levels is shown in Table 3.

Table 3

Breakdown of Job Mean Flow Time
(for FCFS)

Utilization Level	Mean Batching Time (Minutes)	Mean Queueing Time (Minutes)	Mean Batch Processing Time (Minutes)	Mean Flow Time (Minutes)
50%	237.5	30.2	26.5	294.2
70%	169.2	54.9	26.5	250.6
90%	129.5	167.6	26.5	323.6

The 70% utilization level provides the best compromise between the batching and the queueing times, thereby yielding the lowest values of MFT. Since the tendency of the average batching time to decrease and the average queueing time to increase with increase in utilization levels is independent of the flow allowances, the "V" shape of the curve is retained for all values of F. As shown later, the *extent* of increase in queueing times does depend upon the individual dispatching procedure.

The arguments stated above partially explain the behavior of the MT, the PT and the SDT curves at different levels of utilization, especially when the due-dates are set tightly. Recall that the due-date setting procedure incorporates the *expected* batching time which explains the fact that the MT curves are flatter than the MFT curves. However, at 50% utilization, since the interarrival times not only have a large mean value but a large

variance as well (since the interarrival times follow an exponential distribution), the variance in the arrival times and therefore the due-dates of jobs within a batch is high, which in turn implies that the variation between the realized batching time and the expected batching time is high. Consequently, some jobs (which would be the initial jobs in their batches) tend to become tardy. The number of such jobs is, however, quite small, which explains the low values of PT at 50% utilization level. The variation between the realized and the expected values of batching time also explains the large variance in tardiness values as depicted by the SDT curves. As the level of utilization increases, the variance in the interarrival times decreases, which results in smaller discrepancy between the realized and the expected values of the batching time. However, at 90% utilization, the increase in queueing times lead to larger values of tardiness and larger values of PT and SDT as well. An increase in F leads to the crossover phenomenon since large flow allowances not only compensate for the increased queueing time, but at high utilization levels the smaller variance in the interarrival times ensures closer adherence to the expected values of the batching times incorporated in the due-dates. This explains the fact that the values of PT and SDT decrease rapidly with increase in F. The above discussions also explain the fact that the change in the slope of the MT curve in the 50%−70% range with an increase in F is more gradual compared to the change in the slope of the curve in the 70%−90% range.

Evaluation of the Dispatching Procedures

Under the TWK due-date setting rule, BB and MYOP perform uniformly well for all four performance measures across all utilization levels and flow allowances. It appears that, unlike some of the procedures which are optimum for the static problem but which perform rather poorly under a dynamic environment [For example, see Chand (1982) for a discussion on the degradation in the performance of the Wagner-Whitin algorithm when it is applied in a rolling-horizon environment], the effectiveness of BB for the static problem is extended to the dynamic case as well. The effectiveness of MYOP, which is essentially a single-pass heuristic, is, however, surprising. Recall that the BB procedure implements MYOP for determining the initial solution. An analysis of the solutions generated by BB for the static problem has indicated that the initial solution provided by MYOP is quite often optimal; whenever it has been improved upon, the difference in the solution values have been marginal, generally in the order of one percent or less. RNC appears to be quite effective for due-date based performance measures. However, it yields high values of mean flow time at low utilization levels and/or high flow allowances; its performance for this measure is also very sensitive to the actual flow allowance factor used for setting job due-dates. As expected, WBPT exhibits several 'SPT-like' properties such as low mean flow time, low proportion of tardy jobs and high tardiness variance.

However, note that the TWK rule presupposes that the due dates are set deterministically when the jobs arrive and the same flow allowance is used for all jobs. To ascertain the robustness of these dispatching procedures with respect to variability in the values of F, another set of simulation runs was conducted, the results of which are shown in Tables 4 and 5 for the measures of mean flow time and mean tardiness respectively. In this set the flow allowance factor F was assumed to be uniformly distributed between 2 and 20; the due date was determined from Equation (4) using the sampled value of F. In this due-date setting procedure (hereafter termed the *"random"* due-date setting procedure), the correlation between the arrival time of a job and its due date is greatly reduced. [The conceptual difference between the TWK and the random due-date setting procedures lies in the control exercised by the manufacturing function in assigning job due-

Table 4

Mean Flow Time

Dispatching Rule	Utilization Level		
	50%	70%	90%
FCFS	294.2	250.6	323.6
RMDD	295.2	246.3	307.1
MYOP	293.7	244.7	302.0
MONT	291.4	241.7	282.8
WBPT	287.9	236.3	262.4
REDD	295.5	247.3	311.8
NC	292.2	243.3	295.0
RNC	291.0	241.1	290.2
BB	291.0	241.2	290.4

Table 5

Mean Tardiness

Dispatching Rule	Utilization Level		
	50%	70%	90%
FCFS	44.65	29.64	58.03
RMDD	42.65	24.28	32.83
MYOP	41.44	24.42	23.13
MONT	42.40	23.64	45.13
WBPT	43.16	26.84	58.16
REDD	42.60	24.07	36.06
NC	40.96	21.24	18.89
RNC	41.11	21.56	18.42
BB	41.11	21.56	18.36

dates; in the latter case the due-dates are assumed to be set exogenously.] It is evident from Tables 3 and 4 that the BB rule continues to perform effectively, especially at high utilization levels. However, MYOP is consistently dominated by RNC and NC for both performance measures.

Finally, a comment will be made about the relative computational attractiveness of the various rules. All the rules except BB are essentially list processing algorithms. BB is exponential and hence is used operationally with a computational or iteration time trap. If the optimal solution is not found within this trap, the best solution found is used. Given the large size of the experiment used in this research, the simulation was carried out on an AMDAHL V8 computer using SIMAN and FORTRAN programs. The time trap for BB was 5 seconds. The simulation programs incorporating BB, MYOP and RNC were subsequently downloaded on a standard IBM-XT microcomputer with 640 kilobytes of core

memory, and the execution times of these three procedures for solving static problems (which were generated during the simulation run) were monitored. The number of jobs in these static problems varied between 3 and 75. The average time taken to solve a static problem was found to be less than 3 seconds for all three procedures. While MYOP and RNC were able to schedule consistently within 4 seconds, the maximum execution time for BB was 10 seconds.

5.0 Summary and Conclusions

This paper extends the previous research on single-machine scheduling to the case where jobs need to be processed in batches with a constant sequence-dependent changeover time. For the static mean tardiness problem, a solution procedure based on implicit enumeration is proposed. For the dynamic scheduling problem involving implementation of scheduling decisions in real-time, nine dispatching procedures are presented and evaluated.

This research highlights two issues of critical importance for an automated manufacturing center or a traditional job shop where job batching and sequence dependent setup times occur. Firstly, the existence of an optimum utilization level. The results of this study indicate that for a given set of job and batching characteristics and for a given flow allowance factor, the mean flow time of all jobs is minimized by operating at a utilization level which effects the best compromise between the batching and the queueing times of the individual jobs. Such a utilization level is also optimal for due-date based scheduling measures when the flow allowances are small. In this study, for the given data set and the three utilization levels investigated, operating at 70% utilization optimized mean flow time as well as the three due-date based scheduling measures for all flow allowances less than or equal to 10 across all dispatching rules.

Secondly, this study suggests that, for such a manufacturing environment, serious consideration should be given to the dispatching rules BB, MYOP and RNC, for their high expected system performance, and the ability to use them on today's microcomputers for real-time scheduling.

References

1. Afentakis, P. (1985), "An Optimal Scheduling Strategy for Flexible Manufacturing Systems", Working Paper # 85-012, Dept. of Industrial Engineering and Operations Research, Syracuse University, Syracuse, NY.

2. Akella, R., Y. Choong, and S. B. Gershwin (1984), "Performance of Hierarchical Production Scheduling Policy", *IEEE Transactions on Components, Hybrids and Manufacturing Technology*, Vol. CHMT-7, No. 3, 225-240.

3. Baker, K. R. (1974), *Introduction to Sequencing and Scheduling*, John Wiley and Sons, New York, NY.

4. Baker, K. R. and J. B. Martin (1974), "An Experimental Investigation of Solution Algorithms for the Single-Machine Tardiness Problem", *Naval Research Logistics Quarterly*, Vol. 21, 187-199.

5. Baker, K. R. and L. E. Schrage (1978), "Finding an Optimal Sequence by Dynamic Programming: An Extension to Precedence-Related Tasks", *Operations Research*, Vol. 26, 111-120.

6. Baker, K. R. and J. M. W. Bertrand, (1982), "A Dynamic Priority Rule for Sequencing Against Due-Dates", *Journal of Operations Management*, Vol. 3, No. 1, 37–42.

7. Baker, K. R. and J. J. Kanet (1983), "Job Shop Scheduling with Modified Due-Dates", *Journal of Operations Management*, Vol. 4, No. 1, 11–22.

8. Baker, K. R. (1984), "Sequencing Rules and Due-Date Assignments in a Job Shop", *Management Science*, Vol. 30, 1093–1104.

9. Chand, S. (1982), "A Note on Dynamic Lot Sizing in a Rolling-Horizon Environment", *Decision Sciences*, Vol. 13, No. 1, 113–119.

10. Chang, Y., R. S. Sullivan and U. Bagchi (1984), "Experimental Investigation of Quasi-Real-Time Scheduling in Flexible Manufacturing Systems", *Proceedings of the First ORSA/TIMS Special Interest Conference on Flexible Manufacturing Systems: Operations Research Models and Applications*, Ann Arbor, MI.

11. Conway, R. W., W. L. Maxwell and L. W. Miller (1967), *Theory of Scheduling*, Addison-Wesley, Reading, MA.

12. Dar-El, E. M. and S. C. Sarin (1984), "Scheduling Parts in FMS to Achieve Maximum Machine Utilization", *Proceedings of the First ORSA/TIMS Special Interest Conference on Flexible Manufacturing Systems: Operations Research Models and Applications*, Ann Arbor, MI.

13. French, S. (1982), *Sequencing and Scheduling – An Introduction to the Job-Shop*, Ellis Harwood, Chichester, U. K.

14. Graves, S. (1981), "A Review of Production Scheduling", *Operations Research*, Vol. 29, 646–675.

15. Groover, M. P. (1980), *Automation, Production Systems and Computer-aided Manufacturing*, Prentice-Hall, Englewood Cliffs, NJ.

16. Jones, A. and C. J. McLean (1986), "A Proposed Hierarchical Control Model for Automated Manufacturing Systems", *Journal of Manufacturing Systems*, forthcoming.

17. Karmarkar, U. S., S. Kekre and S. Kekre (1984), "Lot-sizing in Multi-Item, Multi-Machine Job Shops", Working Paper # QM8402, University of Rochester, Rochester, NY.

18. Lin, L. S. and C. Y. Joseph Lee (1984), "The Scheduling Problem in Flexible Manufacturing System", *Proceedings of the First ORSA/TIMS Special Interest Conference on Flexible Manufacturing Systems: Operations Research Models and Applications*, Ann Arbor, MI.

19. Montagne, E. R. (1969), "Sequencing with Time Delay Costs", *Industrial Engineering Research Bulletin*, Arizona State University.

20. Morton, T. E. and R. V. Rachamadugu (1982), "Myopic Heuristics for the Single Machine Weighted Tardiness Problem", Technical Report # CMU-RJ-TR-83-9, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

21. Panwalker, S. S., R. A. Dudek and M. L. Smith (1973), "Sequencing Research and the Industrial Problem", in *Symposium on the Theory of Scheduling and its Applications*, edited by S. E. Elmaghraby, Springer, New York, NY.

22. Picard, J.-C., and M. Queyranne (1978), "The Time-Dependent Traveling Salesman Problem and its Application to the Tardiness Problem in One-Machine Scheduling", *Operations Research*, Vol. 26, No. 1, 86–110.

23. Rachamadugu, R. V. (1985), "Some Properties of Weighted Tardiness Problems", Working Paper # 425, Graduate School of Business Administration, University of Michigan, Ann Arbor, MI.

24. Rachamadugu, R. V., N. Raman and F. B. Talbot (1986), "Due-Date Based Scheduling in a Flexible Manufacturing System (The ATS)", Working Paper, National Bureau of Standards, Gaithersburg, MD.

25. Raman, N. (1985a), "A Simulation Model of the Automated Manufacturing Research Facility", Working Paper # NBS-GCR-85–498, National Bureau of Standards, Gaithersburg, MD.

26. Raman, N. (1985b), "A Survey of the Literature on Production Scheduling as it Pertains to Flexible Manufacturing Systems", Working Paper # NBS-GCR-85–499, National Bureau of Standards, Gaithersburg, MD.

27. Santos, C. and M. J. Magazine (1985), "Batching in Single Operation Manufacturing Systems", Working Paper, University of Waterloo, Waterloo, Canada.

28. Sen, T. and S. K. Gupta (1984), "A State-of-Art Survey of Static Scheduling Research Involving Due Dates", *Omega*, 63–76.

29. Shanker, K. and Y. J. Tzen (1985), "A Loading and Dispatching Problem in a Random Flexible Manufacturing System", *International Journal of Production Research*, Vol. 23, 579–595.

30. Simpson, J. A., R. J. Hocken and J. S. Albus (1982), "The Automated Manufacturing Research Facility of the National Bureau of Standards", *Journal Of Manufacturing Systems*, Vol. 1, No. 1, 17–32.

31. Stecke, K. E. and J. J. Solberg (1981), "Loading and Control Policies for a Flexible Manufacturing System", *International Journal of Production Research*, Vol. 19, 481–490.

32. Talbot, F. B., J. H. Patterson and W. V. Gehrlein (1986),'A Comparative Evaluation of Heuristic Line Balancing Techniques', *Management Science*, forthcoming.

33. Wee, T. S. and M. J. Magazine (1981),'An Efficient Branch and Bound Algorithm for Assembly Line Balancing – Part 1: Minimize the Number of Work Stations', Working Paper # 150, University of Waterloo, Waterloo, Canada.

THE INTERACTION BETWEEN DESIGN AND SCHEDULING IN REPETITIVE MANUFACTURING ENVIRONMENTS

W.L. Maxwell, J.A. Muckstadt,
P.L. Jackson, R.O. Roundy

School of Operations Research
and Industrial Engineering
Cornell University
Ithaca, New York 14853

Effective engineering design must simultaneously focus on both the products and the manner in which they are manufactured, assembled and distributed. The scientific and popular literature contains literally thousands of papers that address portions of these engineering design problems. Over the past decade there have been substantial advances in software and analysis tools which have resulted in effective commercially available Computer-Aided-Design (CAD) systems, some Computer-Aided-Manufacturing (CAM) systems and improved Manufacturing-Resource-Planning (MRP) systems. New industrial engineering techniques have evolved for planning and scheduling computer control of machine tools, machining centers, and, in some cases, whole factories. New material handling technology has also changed and improved. Concepts such as Flexible-Manufacturing-Systems (FMS) and Group-Technology (GT) have been proposed and implemented in some instances.

Impressive applications of joint product and process design now exist in the United States such as at IBM's Lexington facility, GE's Appliance Park in Louisville, and Hughes Aircraft in Los Angeles. Additional large and small scale facilities will soon be in operation ranging from GM's new large Saginaw Gear plant to Emerson Electric's relatively small dishwasher motor facility. All of these systems differ from a physical standpoint because the nature of the products and processes are not identical. Nonetheless, there are important similarities as well. All employ information technology that permits essentially real-time knowledge of the status of the system. Much of this knowledge is acquired through a variety of sensors. This knowledge is used to identify and, in some cases, alter faulty processes or components to insure overall high quality. However, none of these systems adequately uses information for real-time resource scheduling. Neither have the product and process characteristics been designed considering that these characteristics greatly affect the ease of scheduling a factory's operation. The objective should be to design effective and competitive manufacturing systems as well as to design quality products that meet customer needs. To do this properly, the economic effects of product and process design on the ease of scheduling and dispatching resources must be taken into account. Otherwise, unanticipated work-in-process (WIP) inventories, lower than estimated throughput and capacity utilization, and higher unit costs will certainly result.

The question then is "What attributes should products and processes have so that schedules can be created that generate high throughput, low WIP and high capacity utilization?" There are some obvious attributes such as low variance in processing times and very high process yields. Even if careful manufacturing systems engineering has been done to achieve these goals and if attention has been focused on combining, simplifying and, when possible, eliminating tasks, it is still possible to have an ineffective system due to the difficulty of dynamically scheduling a facility as product mix and volumes deviate from initial marketing estimates.

The purpose of this paper is to show how product and process design could affect scheduling policy. In particular, we will examine how design affects material flow, WIP and throughput by simplifying the complexity of scheduling in a certain type of factory. The factory we have in mind is focused on the production of a relatively small number of high volume products. The demand for each final product is not known precisely and could vary substantially over time. The process technology is planned such that each of the key components in all final products will be produced on common equipment. Common equipment has been selected because of the expense of dedicated equipment and the uncertainty of product life and the demand rate over an extended period of time for each individual product. Furthermore, the components are assumed to flow through the facility in the same manner, that is, the components all visit the machines in an identical sequence. In essence we are assuming that the components are quite similar in terms of form and function but may differ in their geometric characteristics such as their diameter or length or perhaps the number and depth of holes drilled, slots milled, gears cut, etc. Examples of such systems are found in the manufacture of car axles, computer boards and components for air tools.

The scheduling concepts presented in Section IV depend heavily on the nature of the problem environment we have described. We will be making more specific assumptions as we proceed. The purpose here is to discuss the relationship between product and process design and scheduling policy in the type of repetitive manufacturing system we have specified. Different environments require different scheduling systems from the ones we discuss. Our goal is only to demonstrate that these relationships must be taken into account in the design stage no matter what type of product or manufacturing system is under consideration.

II. PREVIOUS OR-BASED PLANNING AND SCHEDULING RESEARCH

Since the time that OR emerged as an engineering discipline, researchers have been studying various aspects of planning and scheduling problems. (See texts by Baker [1], Johnson and Montgomery [4], Peterson and Silver [8] and Conway, Maxwell and Miller [2] for overviews of the types of models and policies that have been considered.) Recently researchers have pointed out that a modeling structure that considers the interrelationships of system design through shop floor scheduling is necessary (see Maxwell, Muckstadt, Thomas and VanderEecken [6]). These

researchers have demonstrated that a comprehensive modeling structure is needed when designing a system so that inconsistencies are avoided among the models used to design systems, plan production, and control the flow of jobs on the shop floor.

Most of the models in the literature do not address the important question "What is the goal of the model in the planning/scheduling framework?" Consequently, there are a number of flaws in the existing models that have been created and studied at length during the past two decades.

The first weakness is that the problem environment is often assumed to have a fixed, finite horizon with all data, such as the number, processing times, yields and due dates of all jobs, deterministically known at the beginnning of the horizon. Consequently, the dynamics of the environment are totally ignored. To get around this difficulty, a rolling horizon is often used. Nonetheless, the effects of the choice of the horizon length is often unknown. Furthermore the need for complete specification of data for the timing of and duration of all future events within the horizon distorts the estimate of the actual system throughput, WIP and cost. This is particularly a problem at the point in time in which the system is being designed. Generally the information available at the design stage is not precise.

The second major weakness of the models is that they are not linked together in either a feed forward and/or feedback sense, especially models for the design, planning, and scheduling phases.

The third major weakness is that planning and scheduling models do not interact in the sense of considering the potential effect of stochastic events or random variation in event durations.

The model that we will discuss for the repetitive manufacturing environment contains some of these weaknesses as well; however, it does address key issues and avoids some of these weaknesses. It is similar in spirit to the models presented by Graves, et al. [3] and Maxwell and Muckstadt [5].

III. THE DESIGN PROBLEM

The team of product and manufacturing engineers for the repetitive manufacturing environment will specify

a) the nature of products - including the bill of materials, the routing, setup and run times per piece per operation and the part-process yield,

b) the manufacturing system - including the quantities of machine tool types and their location on the shop floor, tooling, fixturing, and processing and assembly methods,

c) the information system - including the control systems on the shop floor that permits real-time understanding of the system's status and dynamically reallocating resources to respond to that status, and

d) the material flow rules - including production and batch transfer sizes.

Traditionally design has concentrated primarily on feasibility and cost. Feasibility implied that the proper amounts of resource would be available to produce the desired average required throughput. The choices of the exact process steps and manufacturing technology was driven by the desire to minimize capital investment and the variable cost of labor and material.

The cost of capital has risen, product lives have shortened, the variety of products has increased, and the complexity, technology and cost of manufacturing systems has increased; consequently, other attributes have become important. These include manufacturing and design lead times, setup and run times, WIP levels, machine utilization, and true throughput capability. These are highly interactive factors. As setup times increase, lot-sizes increase thereby lengthening manufacturing lead times, increasing WIP and perhaps reducing throughput rates. An engineering team must examine the effect of design on these factors. For example, different types and varieties of tooling and fixturing should be considered so that these set-up times are made as small as possible thereby increasing scheduling flexibility. We have also seen systems where design goals have specifically stipulated that the number of tooling types is to be no greater than a given value and that commonality of fixtures and carriers is to be maximized. The result of this type of design philosophy not only reduces the obvious costs but dramatically eases the pressures of scheduling the shop and results in improved logistics performance.

IV. SCHEDULING IN A REPETITIVE MANUFACTURING ENVIRONMENT

Let's now state the characteristics of a repetitive manufacturing environment in a more formal manner. The manufacturing system consists of workcenters in which products are produced. Jobs are released to workcenters that specify both the product to be produced and the production quantity. Each released job has a defined start and completion time for each operation on the appropriate machine. The time period over which a specific job occupies these machines is called a job slot. Each of these concepts is described more completely in the following paragraphs.

The manufacturing system consists of a set of workcenters, which we denote by W. Each workcenter $w \in W$ consists of a non-empty set $m(w)$, whose elements are identical interchangeable machines.

Let P be the set of products produced in the system. Each product $p \in P$ has an associated graph $G_p = (N_p, A_p)$, where N_p is a set of nodes corresponding to the manufacturing operations required to produce the product and A_p is a set of arcs representing a precedence relationship of operations and the movement of WIP inventory. For each $n \in N_p$ there is exactly one workcenter in which the operation is completed, which we denote by $w(n)$.

We assume the products are partitioned into u classes in the following manner. All the products p in class j, denoted by P_j, have identical graphs G_p. Let $M(j)$ represent the set of machine centers required to make the products in class j. We assume $M(j) \cap M(i) = \phi$, $i \neq j$. The manufacturing facility is assumed to operate so that all products $p \in P_j$, $j = 1,\ldots,u$, are produced following a common routing and are produced on machines that are used to make no products in other classes. This partitioning occurs in systems using the group technology concept.

Since resources needed to make products in separate classes are different, we will concentrate on one class. Furthermore, since $G_p = G_{p'}$ for all $p,p' \in P_j$, there is no need to consider subscripts on either classes or product graphs. So let $G = (N,A)$ represent the graph for the class.

We make one additional assumption concerning the products in a given class. The length of time to complete an instance of each job on each machine is the sum of its setup time plus the product of the lot size and the run time per unit. Let $\ell(p,n)$ be this job length for product p in machine center $w(n)$ of operation n. We assume that the lot sizes are chosen and run times and setup times are engineered so that $\ell(p,n) = \ell(p',n)$ for all products p and p' in the same class and for all $n \in N$. Consequently we are assuming that the duration of each instance of the jobs is the same from product to product on a given machine. If the setup times are small compared to the run times for a minimum sized lot (which may result due to the design of a carrier), then there will be a wide range of feasible $\ell(\cdot,\cdot)$ satisfying this property. We feel that a design goal should be to engineer setup times and run times so that these lengths can be made approximately equal.

The effect of these assumptions is that we no longer need to be concerned with the particular products that are being produced since job duration is product independent.

Let us now focus on developing a schedule in a repetitive manufacturing environment. A schedule is simply a specification of a Gantt chart indicating the times when each job is executed in each workcenter. Individual instances of jobs correspond to a particular product with a given start and completion time on each machine. This period of time is

the job slot and the graph of the job slots is the Gantt chart. Each job slot in the Gantt chart is either committed to the production of a particular type of product or is uncommitted. When customer orders are received, uncommitted job slots are assigned to complete the desired production. A customer order consists of a request for different products in various quantities. This order is then expressed in terms of job slots. The order is then scheduled into uncommitted job slots consistent with the order's due date.

Since we in essence are assuming that only one job type exists, the schedule we propose to follow is called a cyclic schedule. A cyclic schedule is specified by a cycle length, C, and an assignment of every node $n \in N$ to a start time $S(n)$ and a machine $M(n)$ with the properties:

1. A proper machine is used for each operation (node)
   $M(n) \in m(w(n)) \ \forall \ n \in N$.

2. Operations on the same machine do not overlap
   if $w(n_1) = w(n_2)$
   and $M(n_1) = M(n_2)$
   and $n_1 \neq n_2$
   and $n_1, n_2 \in N$,
   then the intervals $I_1 = [(S(n_1) + r(n_1)) \bmod C)$
   and $I_2 = [(S(n_2) + r(n_2)) \bmod C)$
   have no point in common, where $r(\cdot)$ is the processing time on the appropriate machine for the given operation.

3. The cycle length, C, cannot be less than the largest amount of processing time assigned to any machine of any workcenter;

$$C \geq \max_{w \in W} \max_{c \in m(w)} \left(\sum_{n \ni M(n) = c} r(n) \right).$$

Property (3) must hold if property (2) holds, so the properties of a cyclic schedule rest on property (2).

V. AN EXAMPLE OF THE DESIGN TRADEOFFS

The purpose of this section is to illustrate the consequences of alternative designs for repetitive manufacturing systems on performance measures such as throughput rates, WIP, cycle times and flow times. We also show that by increasing processing times we can in some instances actually increase system performance.

We begin by considering an example manufacturing system consisting of four workcenters; a turning, a machining, a heat treat and a grinding workcenter. The industrial engineers have determined that based on the processing times, as displayed in Table 1, that each work center should contain one machine. This was established by observing that the machining workcenter is the bottleneck with a processing time per job of 9 time units. The desired throughput rate needed to meet projected demand is 1/9, which corresponds to the capacity of the bottleneck. Table 1 also indicates that each job instance has 6 operations and a prescribed routing. For this case the graph G has the following node and arc sets: $N = \{1,2,3,4,5,6\}$ and $A = \{(1,2),(2,3),(3,4),(4,5),(5,6)\}$.

Operation	1	2	3	4	5	6
Workcenter	1	2	1	2	3	4
Proc. time	4	4	2	5	2	3

Workcenter	1	2	3	4
Total time	6	9	2	3

Table 1

Management has also decided that excess inventory should be avoided because the value of each job is quite substantial. In fact, management has asked that schedules be constructed so that each job remains in the facility for the absolute minimum amount of time. Since the sum of the processing times for the individual operations for each job is 20 time units, the minimum achievable flow time is also 20 time units per job. Thus the goal is to construct a cyclic schedule that has no excess inventory. That is, there should not be any inventory in the system at any point in time that is not being worked on by one of the machines.

The schedule for each operation of two job instances for this scenario is displayed in Figure 1. The notation i-j here and in subsequent figures indicates that the machine is working on operation j of job instance i. The minimum flow time schedule for each job is under the constraints given in this Gantt chart. Unfortunately, this schedule requires machining center to be idle periodically. This occurs because a job cannot begin its first operation unless it is possible to complete all operations without delay. As can be seen from the graph in Figure 1, the

cycle time in the machining center must be 11 time units, including 2 time units of idleness, if the minimum flow time is to be achieved. However, this results in a throughput rate of 1/11, which does not meet the required throughput rate of 1/9. We also note that with this minimum flow time the average number of jobs in the system is roughly 1.82 jobs. This was found using the well known equation $L = \lambda W$ with $\lambda = 1/11$ and $W = 20$.

Since it is impossible to achieve simultaneously a flow time of 20 (no excess inventory) and a throughput rate of 1/9, management now wants a schedule that achieves the throughput rate constraint. Many such schedules can be constructed. However, the flow time and corresponding inventory requirements can differ depending on the chosen schedule.

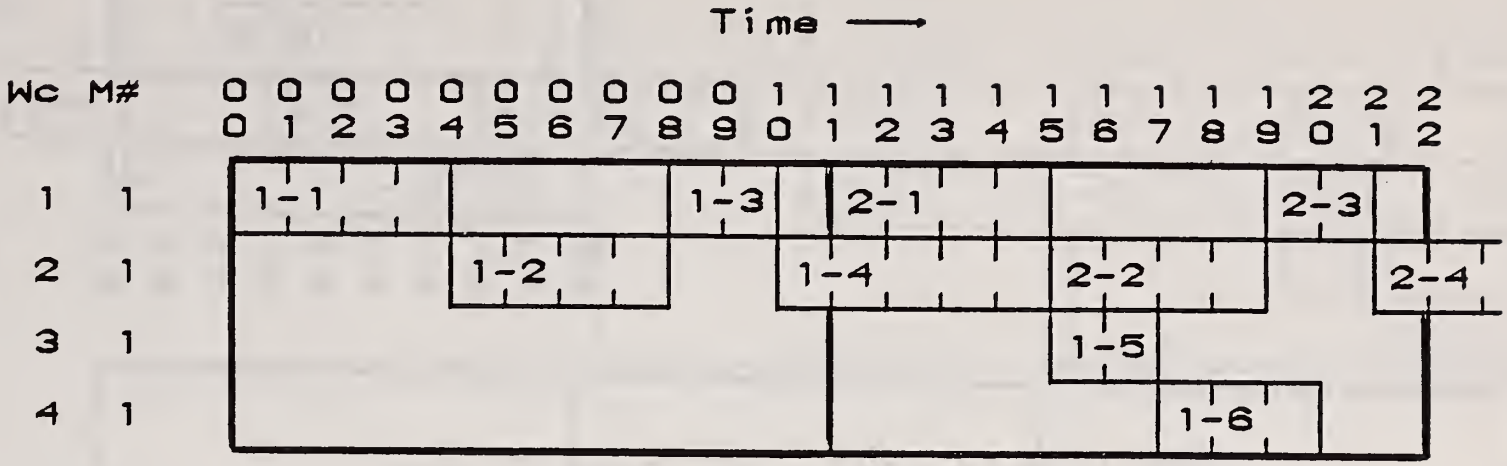
For example, Figure 2 shows a schedule that will permit the desired throughput rate of 1/9 to be achieved. However, the flow time is 60 time units per job thereby increasing the inventory substantially. This schedule will have an average of 60/9 jobs in system, a considerable increase over the minimum 20/11 jobs.

It is possible to construct a minimum flow time schedule given the desired throughput rate of 1/9. Figure 3 displays this schedule whose flow time is 27 time units. Each job is worked on for 20 time units and waits for processing 7 time units. The average number of jobs in the system is now 3 with an average of 20/9 jobs in production and an average of 7/9 jobs waiting to be processed.

Thus given the operation times and the workcenter capacities in this example, there is on average some inventory waiting to be processed. Suppose we want to reduce the flow time so that the waiting time per job (7 time units) is reduced.

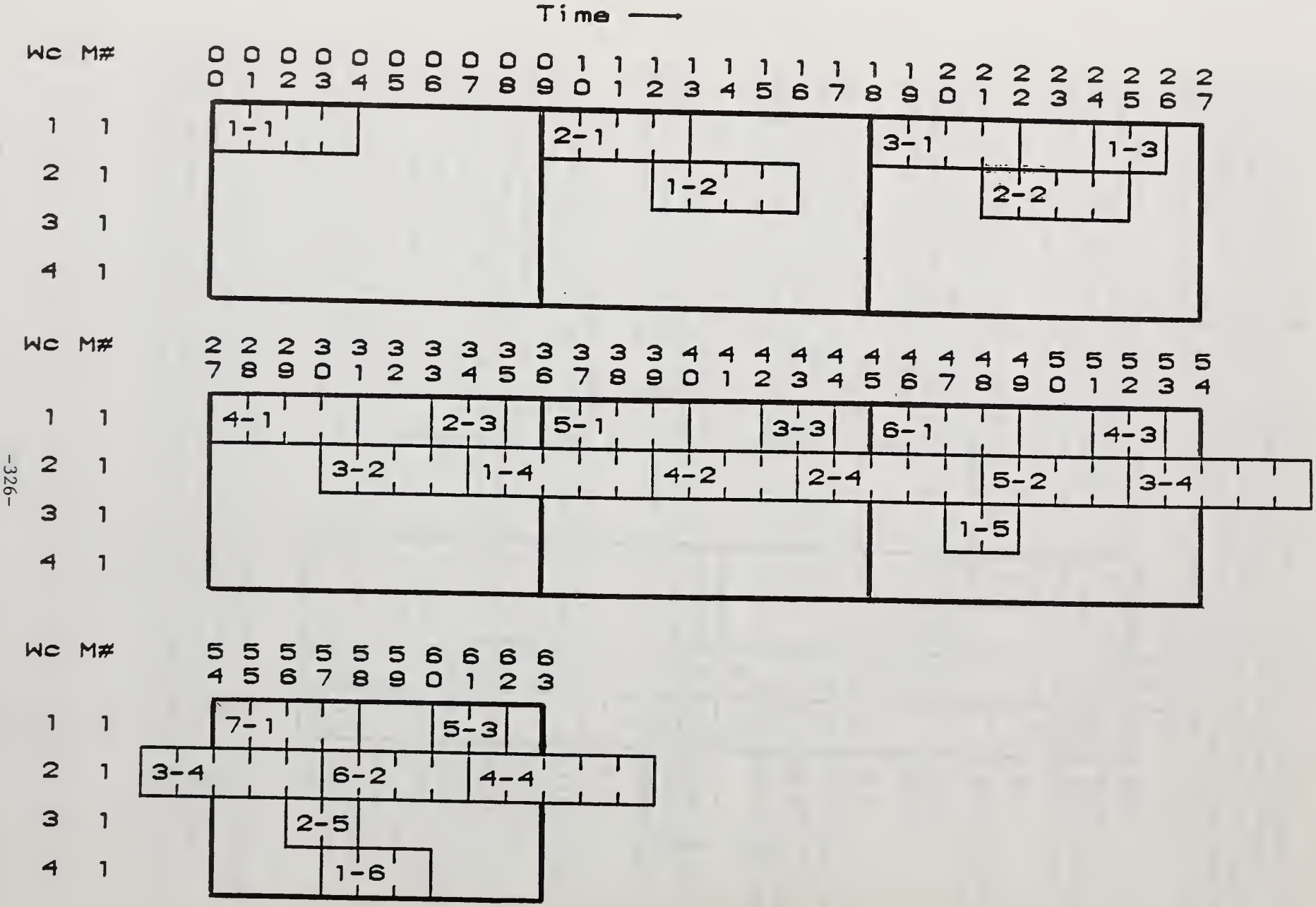
Since the bottleneck workcenter (machining center) requires 9 time units to process each job, it is reasonable to believe that the flow time per job could be decreased by increasing this workcenter's capacity. Suppose a second machine is placed in this workcenter. The resulting minimum flow time schedule for this modified system is displayed in Figure 4. Note that this schedule would not be an easy one to identify without the aid of an algorithm. The flow time is reduced to 25 time units and the average number of jobs awaiting processing is reduced by only 2/9 jobs. Placing additional machines in this workcenter will not further reduce the flow time of each job. Thus it is not possible to eliminate excess inventory by simply adding capacity to the bottleneck workcenter. This happens because each job must return to the turning center after completing operation 2 and because the turning center becomes the bottleneck workcenter.

An alternative way to possibly reduce flow time is to use a different process to complete each job. We assume the manufacturing engineers selected the original process steps because they resulted in a minimum total processing time. This may appear to be the best possible alternative; however, it may not be.



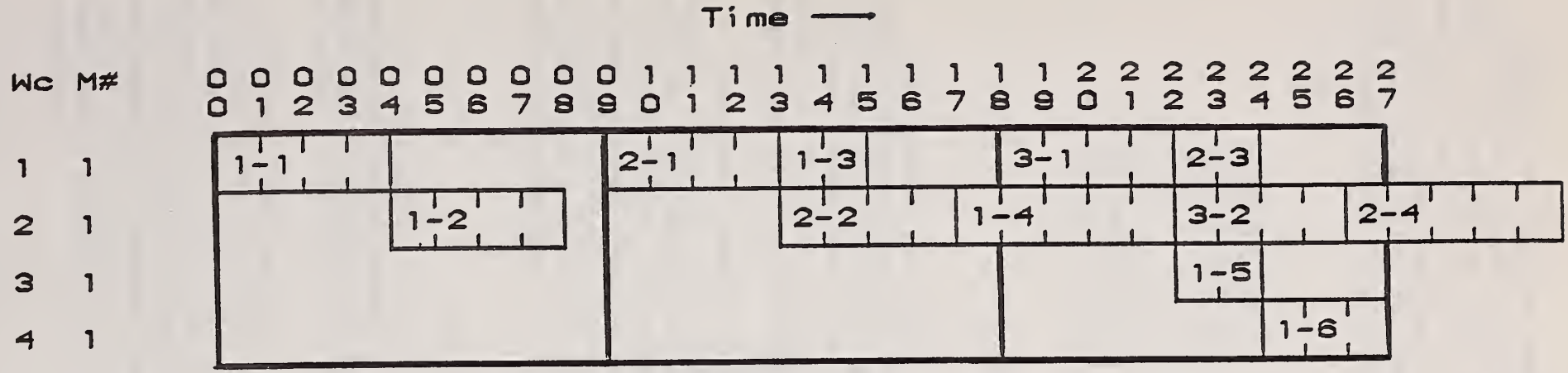
Cycle time = 11 Flow time = 20

Figure 1



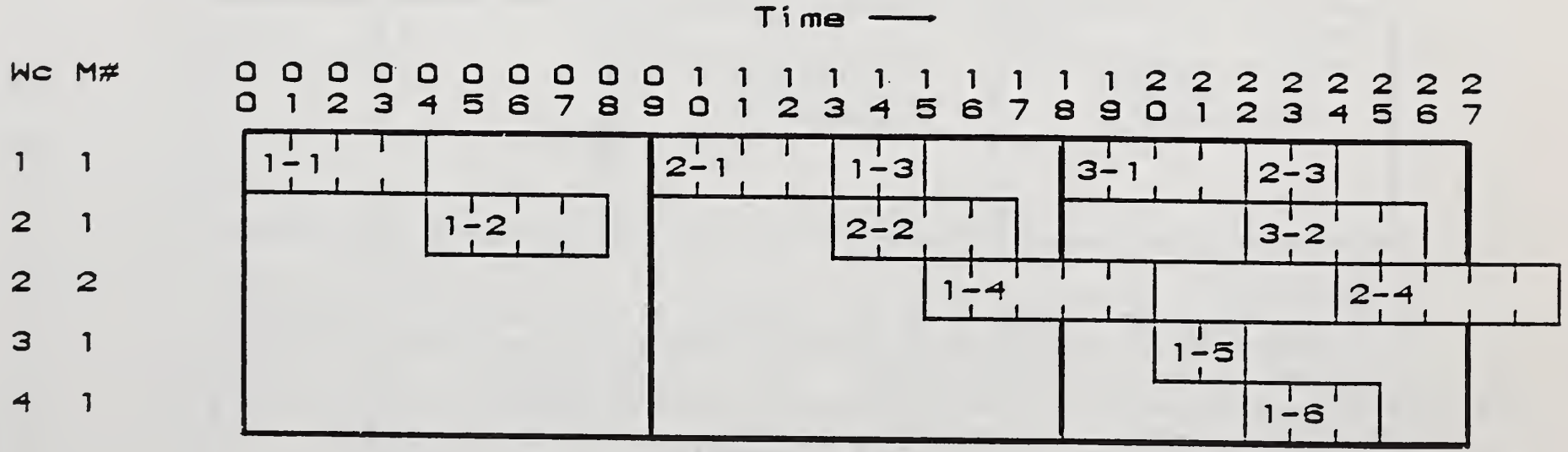
Cycle time = 9 Flow time = 60

Figure 2



Cycle time = 9 Flow time = 27

Figure 3



Cycle time = 9 Flow time = 25

Figure 4

Suppose the required tasks are re-evaluated. It is often possible to accomplish certain metal removal tasks on either a turning or a machining center. We assume the tasks are reevaluated so that operations and operation times are revised to correspond to the data in Table 2. We see that there are two important changes. First, the sum of the processing times has increased to 21 time units, an increase of 5%. Second, each job visits each workcenter only once in the revised routing.

Operation	1	2	3	4
Workcenter	1	2	3	4
Proc. time	7	9	2	3
Workcenter	1	2	3	4
Total time	7	9	2	3

Table 2

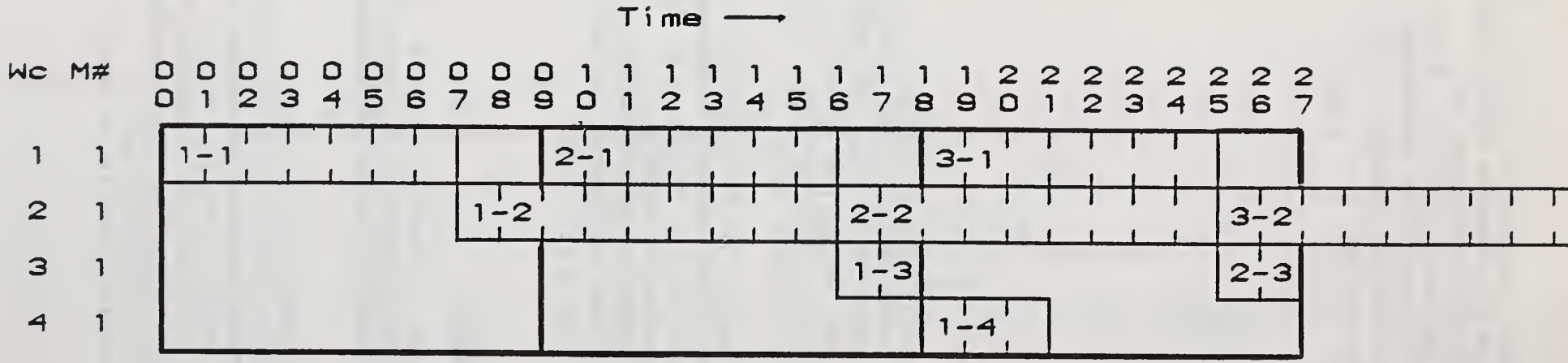
The schedule that results from this revised routing is shown in Figure 5. We see that the flow time for this schedule is 21 time units, the minimum possible flow time. Hence no waiting time is needed and each job flows through the system without delay. Thus there is no excess inventory. Furthermore, since the processing time in the machining workcenter is 9, the throughput rate is 1/9, the desired value.

This simple example problem illustrates some of the effects that process, workcenter and product design can have on key system performance measures. The example clearly demonstrates that careful throught has to go into system design. The relationships among production capacity, processing times, inventory and throughput rate can be quite complicated. We advocate that these relationships be established and used as the basis for designing products, processes and workcenters in repetitive manufacturing systems.

VI. CONCLUDING COMMENTS

In this paper we have discussed the concept of a cyclic schedule. We have shown that by carefully engineering both the products and processes, these cyclic schedules can be effectively employed to improve performance in repetitive manufacturing systems.

We feel that the cyclic scheduling model potentially has many advantages over the classic finite rolling horizon models. The simplicity of these cyclic schedules could be of substantial practical importance when evaluating alternative manufacturing system designs. Some preliminary results have been obtained by Graves et al. [3], Maxwell and Muckstadt [5] and Roundy [7]. However, these works are only a beginning. We encourage researchers and practitioners to examine from both analytic and experimental viewpoints the properties and potential benefits of cyclic schedules.



Cycle time = 9 Flow time = 21

Figure 5

VII. ACKNOWLEDGEMENTS

We are grateful for the invitation from Richard H.F. Jackson, Center for Applied Mathematics and Albert W.T. Jones, Center for Manufacturing Engineering, both from the National Bureau of Standards, to present this material at the Symposium on "Real Time Optimization for Automated Manufacturing Facilities." We are also grateful for their encouragement to present the material in the Proceedings of the Symposium.

REFERENCES

[1] Baker, K.R., Introduction to Sequencing and Scheduling, John Wiley & Sons, Inc., 1974.

[2] Conway, R.W., W.L. Maxwell and L.W. Miller, Theory of Scheduling, Addison-Wesley, 1967.

[3] Graves, S.C., H.C. Meal, D. Stefek, A.H. Zeghmi, "Scheduling of Re-Entrant Flow Shops," Journal of Operations Management, Vol. 3, No. 4, 1983, pp. 197-207.

[4] Johnson, L.A. and D.C. Montgomery, Operations Research in Production Planning, Scheduling, and Inventory Control, John Wiley & Sons, Inc., 1974.

[5] Maxwell, W.L. and J.A. Muckstadt, "A Model for Planning Production in an N Stage System," Technical Report, School of OR&IE, Cornell University, June 1981.

[6] Maxwell, W.L., J.A. Muckstadt, L.J. Thomas, and J. VanderEecken, "A Modelling Framework for Planning and Control of Production in Discrete Parts Manufacturing and Assembly Systems," Interfaces, Vol. 13, No. 6, pp. 92-104.

[7] Roundy, R., "The Combinatorics of Cyclic Schedules," Technical Report, School of OR&IE, Cornell University, March 1986.

[8] Peterson, R. and E.A. Silver, Decision Systems for Inventory Management and Production Planning, John Wiley & Sons, 2nd Edition, 1985.

An Adaptable Scheduling Algorithm for Flexible Flow Lines (Abstract)

Robert J. Wittrock

Manufacturing Research Department
IBM Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Consider a manufacturing line which produces parts of several types. Each part must be processed by at most one machine in each of several banks of machines. An algorithm will be presented, which schedules the loading of parts into such a line. The goal of the algorithm is primarily to minimize the makespan and secondarily to minimize queuing. The problem is decomposed into four subproblems and each of these is solved by using a fast heuristic. Several extensions are made to the algorithm, in order to handle limited storage capacity, large volumes, expediting, and reactions to system dynamics. The algorithm was tested by computing schedules for a real production line, and the results are discussed.

This paper has been submitted to *Operations Research*.

DETERMINING AGGREGATED FMS PRODUCTION RATIOS AND MINIMUM INVENTORY REQUIREMENTS

Kathryn E. Stecke

Graduate School of Business Administration
The University of Michigan
Ann Arbor, Michigan

1. INTRODUCTION

The high level of automation of an FMS allows efficient and flexible simultaneous machining of a variety of part types in unit batch sizes. The operation of these systems is different from the traditional assembly line or job shop situations. The FMS planning, scheduling, and control problems have sometimes similar, but often different counterparts in the conventional manufacturing systems.

Five production planning problems were defined in Stecke [1983] to help an FMS manager set up his/her system in an efficient and productive manner prior to the start of production. Several of these have been addressed previously at various levels of detail. This paper addresses a different two of the five planning problems.

The plan of the paper is as follows. §2 begins by briefly reviewing the planning problems and various solution approaches to date. We discuss how these problems and appropriate solution procedures are different for FMSs as well as how their solutions relate to the FMS scheduling problems that would need to be solved subsequently. §3 suggests solution approaches to determine aggregate production ratios for several relevant operating objectives and associated problems. §4 takes the results of §3 as input into two models to use to help determine operating solutions. A summary and future research needs are provided in §5.

2. PROBLEM DEFINITION

The following five planning problems have to be addressed and implemented in an FMS, periodically and in advance of the start of production of a new or different part mix. Suri and Whitney [1984] call problems like these, second level decisions, to be addressed over a time horizon of several days or weeks.

1. The part types that are to be produced next, and simultaneously over the upcoming period of time, have to be selected.

2. Within each machine type, machines may be partitioned into identically-tooled machine groups. Then each group can perform the same operations. Grouping is useful in that it automatically provides redundancy for breakdown situations; it automatically provides for alternative part routings; it decomposes the tooling problems into smaller problems and makes them easier to solve. However, grouping is not essential and sometimes cannot be performed. The necessary planning functions can be addressed directly in (5) below.

3. The production ratios at which the selected part types should be produced over time are determined.

4. The minimum numbers of pallets and fixtures of different fixture types required to maintain the production ratios need to be determined.

5. The cutting tools of all operations of all of the selected part types have to be loaded into some machine's (one or more) limited capacity tool magazine in advance of production. This determines which machine tools each operation can be performed on during the real-time production of parts.

There are production requirements that usually change over time for a variety of part types. These production requirements are derived either from some forecast of demand or actual customer orders. Depending on many factors, such as system capacity or due dates, for example, usually some subset of the required part types will be chosen to be produced next over the upcoming time period. When the production requirements for some part type(s) are finished, space is freed up in the tool magazines and either one or more part types can be input into the system (if space for all cutting tools can be found) or just the reduced set of part types can be machined (perhaps more pooling can be done). An alternative heuristic to select the part types to be produced next is suggested by Whitney and Gaul [1984]. They partition all part types into batches, and then machine one batch at a time.

The grouping and loading problems (problems (2) and (5)) have been treated at several levels of detail. Queueing networks have been used to characterize appropriate solutions to these and other FMS problems and to provide qualitative or operational insights in Buzacott and Shanthikumar [1980], Cavaillé and Dubois [1982], Dubois [1983], Solberg [1977, 1979], Shanthikumar and Buzacott [1980], Shanthikumar and Stecke [1986], Stecke [1986], Stecke and Morin [1985], Stecke and Solberg [1985], Suri [1983], Suri and Hildebrandt [1984], and Yao [1984], for example. At a detailed level, the various problems were addressed using mathematical programming (Stecke [1983], Berrada and Stecke [1984]), heuristics (Stecke and Talbot [1983], Whitney and Gaul [1984]), and at a less detailed level by Kusiak [1983].

This paper addresses the third and fourth planning problems. In particular, aggregate approaches to help determine appropriate, "optimal" ratios in which a selected set of part types should be produced are suggested in §3. The main, overall system objective that is considered in this paper is to maximize production or system utilization. The equipment is expensive and many FMS

users admit that a high utilization and maximum production is of major concern (i.e., Vought AeroProducts, John Deere, Renault Machines Outiles,...). If due dates are relevant, these would impact other problems, such as part type selection as well as the part input sequence and scheduling procedures. Makespan might also be important, but maximize utilization could help to attain makespan objectives.

Three relevant objectives to determine the production ratios to follow that will help maximize production are considered here. Each would be applicable in a different type of FMS. These various FMS situations and those relevant for consideration here are now described.

Some FMSs produce parts that are required in certain relative ratios. For example, perhaps the system machines many parts or components for later assembly purposes. Then the parts are required in certain, perhaps equal, output ratios of each. These requirements can be translated into operating production ratios in this case, as we shall see in §3.1. There are also interesting part type selection, grouping, loading, part input sequence, and scheduling problems in this case, which are not the subject of this paper.

The systems that are of more interest here are those that machine independent part types. There may be requirements for varying numbers of each, but we are free to determine the relative ratios in which they should be produced. There are several scenarios possible in this case. Some approaches to operate the system are better than others (with respect to maximum utilization).

We first examine the situation in which a set of part types has been selected to be machined, having varying production requirements, with the operating objective of starting and finishing all of these parts at the same time. There are plausible reasons for such an operating decision. Before production begins, all required cutting tools have to find their places in some tool magazines(s). When all requirements are finished, all magazines can then be emptied and the system set up for the next set of part types. This approach tends to minimize the frequency of tool changes. We show in §3.1 how to determine aggregate ratios of parts so that they begin and end all requirements simultaneously. One can see that this approach defines the workload constraints and the bottleneck machine (type). In general, it will neither maximize production nor utilization. However, this may be an appropriate approach for some FMSs, for example, if demand for some parts is dependent and certain relative output ratios are required. These output ratios would be translated into different production ratios. We show how to implement this approach in §4.2

The operation of the FMS is different in §3.2. Aggregate ratios of part types are determined so as to keep the workload per machine on each of the machine types relatively balanced (or unbalanced, if that is applicable). In this case, the operation of the FMS is more flexible. Now, the requirements of some part type are finished first, and all of the planning problems are addressed again, including the determination of production ratios. The planning for set up of the system prior to production is more complicated, but system utilization and production increases. Groups of pooled machines are also considered.

Throughout §3.2, it is seen that by determining ratios to balance workloads, idle time tends to decrease, the amount of buffer space required is less than otherwise, less lead time is required, and inventory requirements are minimized. Some theoretical justification of the latter observation is provided by Shanthikumar and Stecke [1986].

In some systems, parts require one or more refixturings so that cutting operations can be performed on a different surface of the part. In these situations, there are relative ratios predetermined for each fixturing of each part type, but if the types are independent, ratios can again be found. Another situation that can be handled similarly is that of different components being machined in fixed ratios but for different assemblies. The ratios of assemblies can then be determined. These situations are addressed in §3.2.5.

In §4, we use the aggregate results of §3 as input into more detailed models to determine optimal, operating production ratios. We also specify how to solve the related problem of determining the minimum number of pallets and fixtures of different fixture types that are needed to maintain the desired, calculated production ratios.

3. DETERMINING AGGREGATE PRODUCTION RATIOS

We begin by defining in Table I the notation that will be used subsequently.

TABLE I

Notation

i	part types, $i = 1,\ldots,N$
j	machines, $j = 1,\ldots,M$
k	machine types, $k = 1,\ldots,K$
a_i	production ratio of part type i
r_i	production requirements for part type i
n_i	number of pallets required for part type i
p_{ij}	processing time of part type i on machine j
tp_i	total workload of one part of part type i
m_k	number of machines of type k
n	total number of pallets required $= \sum_{i=1}^{I} n_i$

3.1 FINISHING ALL PARTS AT THE SAME TIME

Given the part types that have been selected to be machined simultaneously over some upcoming production period, and each part type's total processing time and production requirements, the problem is to find a set of aggregate production ratios to be followed that allow all part types to finish at the same time.

The aggregate production ratios are obtained simply by solving the following equations for a_i, $i = 1,\ldots,N$.

$$r_1 (tp_1)/a_1 = r_2 (tp_2)/a_2 = \ldots = r_i (tp_i)/a_i = \ldots = r_N (tp_N)/a_N \qquad (1)$$

The situation can be thought of as a static, deterministic, aggregate, minimum makespan-like problem. Travel time, waiting time, and the like are not considered here. The real-time control of production accounts for these. These aggregate production ratios, a_i, serve as guidelines for production. For example, they impact and can be used to help determine the part input sequence. Appropriate scheduling procedures will direct the flow of work through the system. These other considerations, i.e., waiting time,..., are accounted for shortly in determining actual production ratios.

The following simple example illustrates the concepts. Table II contains processing time information and production requirements for two part types on two machine types (mills and drills, say).

TABLE II

Processing Times for Two Part Types on
Two Machine Types with Production Requirements

	Mill	Drill	r_i
PT_1	10'	40'	50
PT_2	20'	10'	100

Substituting the appropriate information into equation (1) and solving, the aggregate production ratios are: $a_1 = 5$ and $a_2 = 6$.

It can be seen that maintaining these relative ratios over time will help to allow the completion of all requirements of both part types at the same

time. This might be a goal of Whitney and Gaul's [1984] batching procedure. Begin a batch, complete its requirements over some time horizon, and then begin the next batch. The frequency of tool changes are minimized. However, production rate is lower using this approach, rather than the different objective approach of §3.2. Hence the total number of tool changes will also be less.

This approach is also applicable if, for example, two of PT_2 are needed for every PT_1. This required output ratio translates directly to the production requirements in Table II of 100 and 50 pieces, respectively.

As another example, if these requirements are for one customer, and the orders need to be shipped together when completed, and there is no area for finished goods inventory, this approach is appropriate also.

Operating the system in this manner defines the workload. In this particular example, the workload unbalance is not too bad. Over time, the drills will be the bottleneck machine type. In §3.2, another approach, applicable for different types of systems, determines the ratios to provide a better workload balance.

Notice that we have not yet accounted for travel time, waiting time, or congestion. Only the aggregate ratios have been determined. These other considerations are accounted for in §4.2, where the ratios are input into other models that determine the actual operating production ratios as well as the minimum number of pallets and fixtures to maintain these ratios. Rarely, these ratios are slightly revised at this next stage. Finally, we note that this procedure generalizes immediately to N part types, M machines, and K machine types.

3.2 BALANCING WORKLOAD PER MACHINE

Given the processing time requirements of each part type on each machine type, the problem is to determine relative ratios at which the part types should be maintained in production, so as to keep the workloads on the machine types balanced.

For different types and sizes of problems, the following suggested solution procedures differ. Also, for larger problems there are multiple "optimal" solutions with respect to balancing, so that other, secondary criteria can be used to determine the ratios to follow. For these reasons, the presentation consists of cases, presented in order of increasing complexity. Examples are used to illustrate. The benefits to be obtained from following the suggested procedures are first demonstrated in §3.2.3, as the situations become sufficiently complex enough to be of interest.

Aggregate production ratios are determined that are to be followed over time. These ratios are input into the more detailed models in §4.2. They are revised, if need be. More usually, the minimum number of pallets and fixtures to maintain these ratios can be found via the procedures of §3.

3.2.1 Two Part Types, Two Machine Types

The simplest situation consists of two part types and two machine types with one machine of each type. Using the notation of Table I, the ratios to balance the workload on both machines over time can be found by solving:

$$p_{11}\, a_1 + p_{21}\, a_2 = p_{12}\, a_1 + p_{22}\, a_2. \tag{2}$$

The solution is:

$$\frac{a_1}{a_2} = \frac{p_{22} - p_{21}}{p_{11} - p_{12}}. \tag{3}$$

Note that the quantity on the left (right) hand side of equation (2) is an aggregate measure of workload over time on machine one (two). Equating the two quantities balances the workload.

If the solution (3) consists of positive a_1 and a_2, these are then the ratios to maintain over time to balance the workload. However, care should be taken to ensure a feasible (all a_i greater than zero) solution to equation (2). In selecting the two part types to be produced, one has to utilize one machine type more, while the other part type has to utilize the other machine type more. Otherwise, a workload balance between the two machines is impossible and a queue has to build and idle time results. This situation would surface in (3) as the ratios, a_1 and a_2, would then relate negatively to each other. Obviously, the processing time requirements on each machine type j cannot be identical.

To illustrate with the data in Table II, the production ratios that balance the workload on the mill and drill are:

$$a_1 = 1 \text{ and } a_2 = 3.$$

Again, we return to this in §4.2, where these aggregate operating ratios are input into more detailed models.

The procedure generalizes immediately to systems containing pools of machines. If there are several machines (m_k) of each type k, then the workload per machine is balanced by solving:

$$p_{11}\, m_2\, a_1 + p_{21}\, m_2\, a_2 = p_{12}\, m_1\, a_1 + p_{22}\, m_1\, a_2. \tag{4}$$

The solution is:

$$\frac{a_1}{a_2} = \frac{p_{22}\, m_1 - p_{21}\, m_2}{p_{11}\, m_2 - p_{12}\, m_1}. \tag{5}$$

To illustrate with the same data of Table II, if there are 2 mills and 4 drills, the relative ratios that balance the workload per machine on each mill and drill are:

$$a_1 = 3 \text{ and } a_2 = 2.$$

The next generalization is to include a third part type.

3.2.2 Three Part Types, Two Machine Types

By including a third part type 3 on the system, the equation to solve to find the production ratios of three part types on two machines is:

$$p_{11}\, a_1 + p_{21}\, a_2 + p_{31}\, a_3 = p_{12}\, a_1 + p_{22}\, a_2 + p_{32}\, a_3. \tag{6}$$

The solution to equation (6) is of the form:

$$a_2 = \frac{(p_{12} - p_{11})\, a_1 + (p_{32} - p_{31})\, a_3}{p_{21} - p_{22}}. \tag{7}$$

In this case, the solution is not a set of ratios. The solution described by (7) is an equation, in particular, a plane. If the problem is well-defined (along the lines described in §3.2.1), then there is an infinite number of solutions that will balance each machine's workload. The problem of how to choose one of these solutions is addressed in §4.

To illustrate, consider the inclusion of a third part type in addition to those of Table II, requiring 10 minutes on the mill and 20 minutes on the drill. These requirements are in reverse to those of part type 2. In this case, the solution to equation (6) is:

$$a_2 = 3a_1 + a_3. \tag{8}$$

Table III contains some possible solutions, all of which balance the workload.

TABLE III

Production Ratios from Equation (8)

a_1	1	1	2	2	1
a_2	4	5	8	7	6
a_3	1	2	2	1	3
n	6	8	12	10	10

If, for example, the number of pallets in the system is the sum of the ratios, we see that there can be several ways to distribute some n pallets among the three part types so as to balance workload. We return to this issue in §4. Notice that although part types 2 and 3 have asymmetric machine requirements, their ratios will never be equal unless type 1 is not produced.

Finally, note that in the case of pooling machines, the equation to solve for the optimal production ratios is:

$$p_{11}\, m_2\, a_1 + p_{21}\, m_2\, a_2 + p_{31}\, m_2\, a_3 = p_{12}\, m_1\, a_1 + p_{22}\, m_1\, a_2 + p_{32}\, m_1\, a_3. \quad (9)$$

One solution to equation (9) is of the form:

$$a_2 = \frac{(p_{12}\, m_1 - p_{11}\, m_2)\, a_1 + (p_{32}\, m_1 - p_{31}\, m_2)\, a_3}{p_{21}\, m_2 - p_{22}\, m_1}. \quad (10)$$

3.2.3 N Part Types, N Machine Types

The procedure described now to find the production ratios is different from those provided in the previous sections. In the present case, N equations are solved for N unknowns. In order to obtain a feasible and meaningful solution (i.e., all $a_i > 0$), an initial, simple check should be made to see that the maximum processing time of each of the N machine types is required by a different part type.

The ratios to balance the workload on the machine types over time can be found by solving the N equations:

$$W = \sum_{i=1}^{N} a_i\, p_{ij}, \quad j = 1, \ldots, N. \quad (11)$$

To illustrate the procedure, consider the aggregate processing time information in Table IV. We want the workload, W of equation (11), to be the same on each machine type:

$$W = 10a_1 + 20a_2 + 10a_3 = 20a_1 + 10a_2 + 30a_3 = 50a_1 + 5a_2 + 20a_3.$$

Since the three equations are dependent, a value for W has to be chosen. The relative ratios remain the same, regardless of the value of W. The selected W merely scales the a_i. Setting W = 100 and solving the three equations simultaneously, we obtain: $a_1 = 1.083$, $a_2 = 3.783$, $a_3 = 1.35$. Doubling each a_i, we obtain: $a_1 = 2.106$, $a_2 = 7.566$, $a_3 = 2.7$. Rounding these values translates to ratios of about 1:4:1 or 2:8:3.

TABLE IV

Processing Times for Three Part Types on Three Machine Types

	Mill	Drill	VTL
PT_1	10	20	50
PT_2	20	10	5
PT_3	10	30	20

Simulating this situation quickly showed that maintaining the ratios of 1:4:1, and including a second PT_3 every other period, both kept the machines balanced and minimized the in-process inventory. Of course, an appropriate part input sequence has to be determined and the calculated production ratios help with this problem also. They provide guidelines to follow. By following the production ratios of 1:4:1.5, several input sequences provided: a balanced workload; minimum work-in-process required; minimum buffer space required; and minimum idle time. In fact, the minimum number of pallets and fixtures of each fixture type required to maintain the ratios was either:

(1, 4, 1) or (1, 3, 2).

Only 6 pallets in total were required. For all other sets of aggregate production ratios that were simulated, queues built up, there was inserted idle time, additional pallets (inventory) were required to keep machines busy, workload was unbalanced, and more buffer space at each machine was required.

3.2.4 N + M Part Types, N Machine Types

A more usual situation is when there are more part types being machined concurrently than machine types. As in §3.2.2, the solution will most often no longer be unique. There could be an infinite number of solutions, but as we shall see, most of these can often be eliminated as either infeasible or undesirable.

The procedure to find the optimal aggregate production ratios is similar to that described in §3.2.3: The N equations (11) are solved for the a_i. However, there could be many solutions. The following example illustrates this situation.

Consider the three-machine system described in Table V. In order for the workload on each machine to be identical, equation (11) provides the three equations:

$$W = 10a_1 + 20a_2 + 10a_3 + 15a_4 = 40a_1 + 10a_2 + 30a_3 + 20a_4$$
$$= 50a_1 + 5a_2 + 20a_3 + 40a_4.$$

TABLE V

Processing Times for Four Part Types on Three Machine Types

	Mill	Drill	VTL
PT_1	10	40	50
PT_2	20	10	5
PT_3	10	30	20
PT_4	15	20	40

Setting W = 100, we can solve for a_1, a_2, and a_3, as functions of a_4:

$$a_1 = 1.7808 - 1.1959a_4$$
$$a_2 = 4.3477 - .739a_4 \qquad (12)$$
$$a_3 = -.4347 + 1.1738a_4.$$

It appears as though there would be an infinite number of solutions. In reality, the feasible set of solutions is small. For any integer value of a_4 greater than one, a_1 is negative (i.e., infeasible). The equations (12) are graphed in Figure 1. For any values of a_4 outside of the interval (.37, 1.49), either a_1 or a_3 is negative. For $a_4 = 1$, $a_1 = .585$, $a_2 = 3.6$, and $a_3 = .739$.

Rounding these values up to integer values, suggests aggregate production rations of: $(a_1, a_2, a_3, a_4) = (1, 4, 1, 1)$. (13)

It can be seen that in this example, any other ratios that would tend to balance workload would be fractional, i.e., $a_4 = .5$.

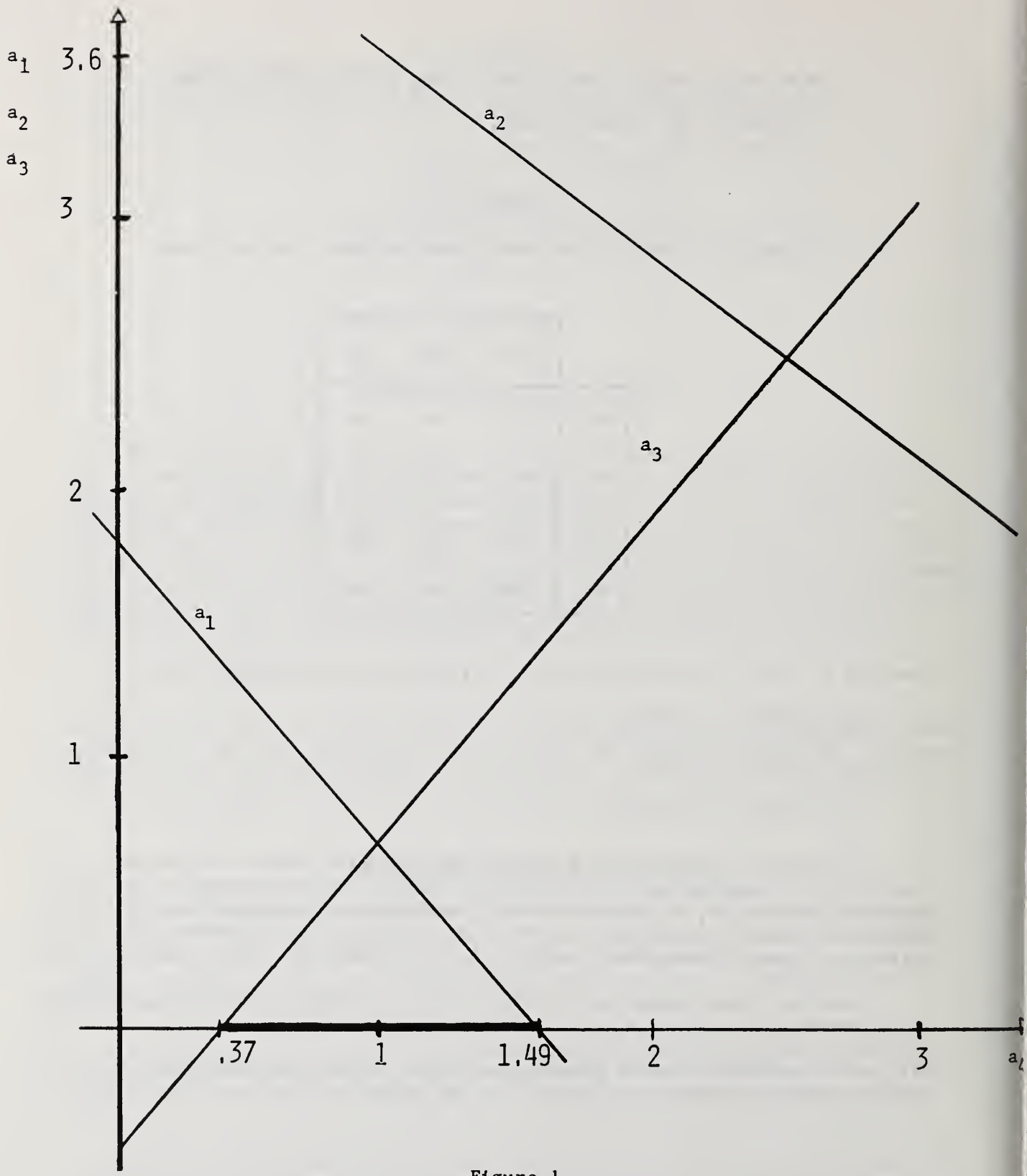


Figure 1

Graph of Feasible Solutions to Equations (12).

Figure 2 is a Gantt chart of one possible scenario. It depicts a flow shop, where each part visits the mill first, the drill second, and the lathe last. Using the aggregate production ratios (13) as guidelines, a good part input sequence was determined to be:

1, 2, 2, 2, 3, 4, 2.

The subscript of each part number in Figure 2 is the pallet/fixture number that is assigned to that individual part. The sequence is periodic and repeats as is. The small boxes (■) indicate the completion of a cycle of the input sequence. Three cycles are shown here.

The production ratios of equation (13) were very useful in the following ways:

1. They were useful as guidelines to help find a good, periodic, input sequence of parts.

2. They helped to find a schedule with very little idle time. Workload is nearly perfectly balanced.

3. The numbers of pallets/fixtures required to maintain these production ratios is exactly the values of the ratios: 1, 4, 1, 1. The total number of pallets required is 7.

4. The amount of buffer space at each machine to hold the WIP inventory is minimal.

The little idle time on the mill in Figure 2 can be decreased even further by following the ratios: (.5, 4, 1, 1). These ratios are closer to the optimal fractions that balance workloads. A part input sequence that provides an even better schedule (less idle time,...) while following these new ratios is:

1, 2, 2, 2, 3, 4, 2; 2, 3, 2, 2, 4, 2.

Again, the procedure generalizes immediately to N machines and N + M part types. The usual situation is that there are more part types than machine types. This could result in several sets of "optimal" aggregate production ratios.

3.2.5 Refixturing

For most types of prismatic parts, after a series of operations are performed, they move off the system to be refixtured. The part is clamped to a different fixture type on a different pallet. The part is then released to the system again and additional cutting and inspection operations are performed on a different surface of the part. Each refixturing in most respects can be treated as a new part type. However, for each part, the production ratios of the refixturings have to remain at one to one. If the end products are independent, aggregate ratios can be found for these that balance the workload. Depending on the numbers of part types and machine types, the appropriate methods described earlier in §3.2 can be applied to find these ratios.

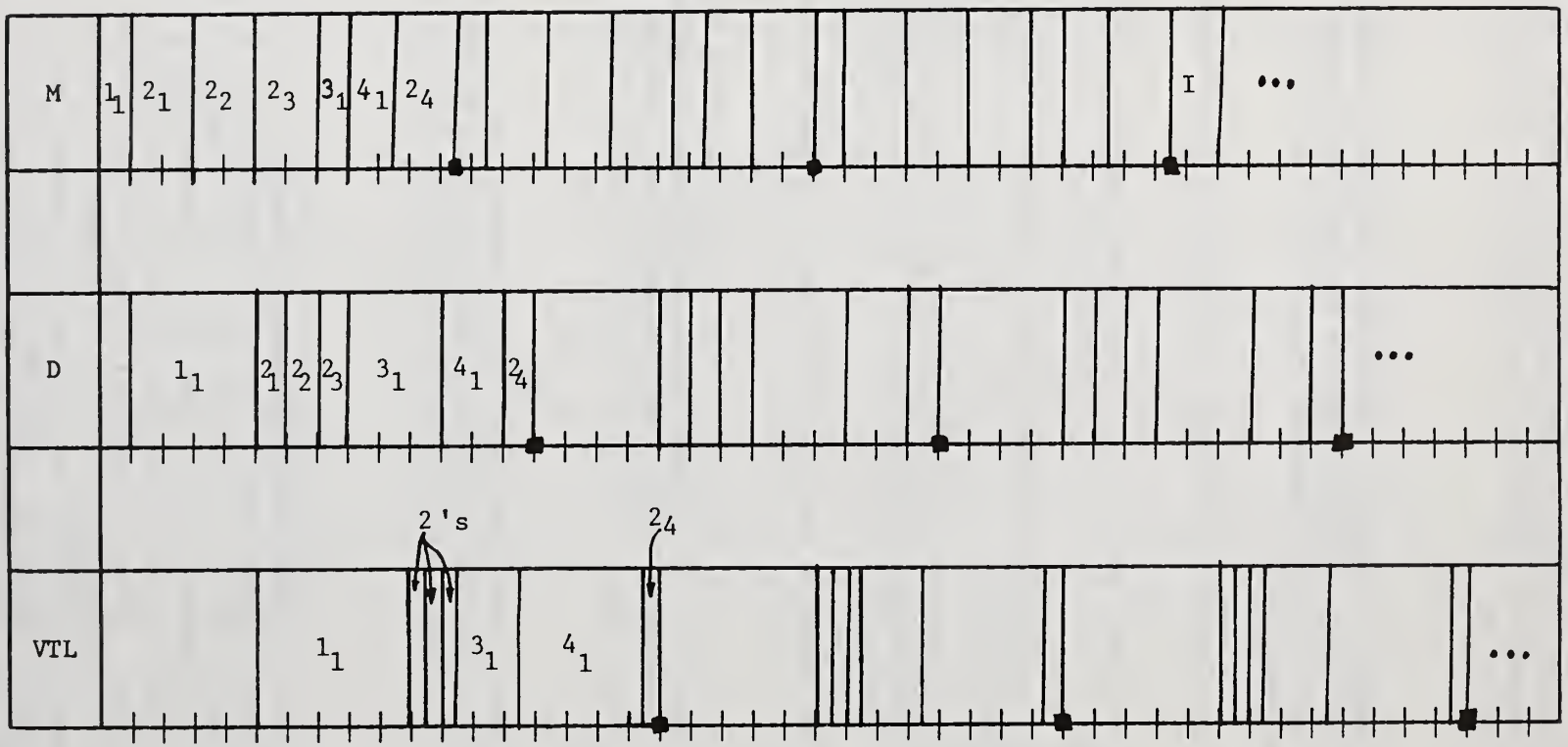


Figure 2

Gantt Chart for a Possible Scenario of the System of Table V.

We again illustrate with an example. The system descibed in Table VI consists of two machine types processing two part types, each of which required a refixturing after passing through the mill and then the drill. In Table VI, PT_{ij} is part type i with pallet/fixture combination j.

TABLE VI

Processing Times for Two Part Types,
Requiring Refixturings, on Two Machine Types

	Mill	Drill
PT_{11}	10	40
PT_{12}	10	30
PT_{21}	20	10
PT_{22}	15	20

Aggregating the processing time information of Table VI and substituting into equation (2), the aggregate production ratios are: $(a_1, a_2) = (1, 10)$. Maintaining these ratios balances the workload. However, in a flow shop situation, and ignoring for the moment the refixturing, set-up, and transportation times while considering the processing and queueing times, only three, rather than ten, fixtures for part type 2 are required to produce at the indicated production ratios. This determination includes waiting time and buffer requirements. When the delays due to transportation and fixturing times are accounted for, a few more pallets will often be required.

4. DETERMINING MINIMUM INVENTORY REQUIREMENTS

The methods of §3 provide aggregate production ratios to follow over time to balance the workload. It has been demonstrated that, in addition, they are useful as guidelines to suggest appropriate input sequences and to minimize inventory requirements. In this section, several of the remaining issues that were mentioned earlier are addressed. Other problems, that remain as future research needs, are addressed in §5.

The usual situation is that there are more part types simultaneously being machined than machine types. Then the ratio problem is similar to that of §§3.2.2 and 3.2.4, where there can be potentially many optimal sets of aggregate production ratios to choose from. In §4.1, some suggestions are offered to help select operating (but still aggregate) production ratios.

In §4.2, we address the problems of determining: (1) actual operating production ratios; and (2) minimum inventory requirements to operate at these ratios. The suggestions of §4.1 serve as input into any of several models

that can help determine both the best ratios at which to operate the FMS and the minimum numbers of pallets and fixture requirements. Both queueing networks and Petri nets are used to determine these. Simulation has been used (see Schriber and Stecke [1986]). Often that much detailed modeling capability is not required. More aggregate or simpler models might be desirable because of the frequency of solution that would be required. However, simulation would be the most frequently used evaluator.

4.1 DETERMINING AGGREGATE OPERATING PRODUCTION RATIOS

When there are more part types than machine types, there are potentially many optimum solutions to the aggregate production ratio problem. After the appropriate equations are solved, feasible intervals for the ratios should be found, as shown in §3.2.4. Because the ratios have to be greater than zero, the intervals can be quite small. Of course if all part types dominate the same machine type, then no feasible ratios can balance the workload per machine.

In some situations, graphical methods are helpful to both determine the ratio intervals and to choose ratios, as in §3.2.4. In particular, a graph is useful for situations in which N + 1 part types are to be produced on N machine types. Otherwise, Tables of feasible combinations can be developed, as shown in §3.2.2, i.e., like Table III.

In any case, there could be a question concerning how to round fractional optimal aggregate ratios up or down to integer values. Fortunately, performance does not appear to be sensitive to small variations in the ratios. In addition, considerations of transportation, fixturing, and queueing time may revise the ratios slightly (more inventory required), but in these situations also, experience has indicated that system performance does not appear to be very sensitive to variations in the ratios.

More specifically, due dates may have been used in the first planning problem to select the part types to be simultaneously machined next. From the feasible sets of ratios, those that best ensure that the due dates are met can be selected. To determine that the due dates can be met, processing time requirements, transportation, queueing, expected down time of the machine tools, for example, have to be considered. Also, real-time control has to occur to monitor continuously the performance to ensure that no part type's due date is in jeopardy. If possible, appropriate action (or reaction) might be taken, in breakdown situations for example, to change the way the system is operated (to change ratios, for example), so as to meet the due dates.

In these types of situations, artificial intelligence might be useful, for the purpose of real-time, continuous monitoring. A rule-based expert system could be developed to propose certain actions to take, if the system state changes drastically (i.e., a machine breakdown). Such a system could be used to choose, update, or change the production ratios as the system changes. Different optimal ratios, all of which tend to balance workload, can be chosen by the expert system, as the system state indicates. The different set of ratios will specify different pallet distributions.

4.2 DETERMINING ACTUAL PRODUCTION RATIOS AND MINIMUM INVENTORY REQUIREMENTS

The aggregate ratios found by the methods of §3 serve as guidelines to help provide input into the models now described to find actual operating ratios.

One model that is useful for these purposes is a multiclass closed queueing network, such as MVAQ. (See Cavaillé and Dubois [1982], Suri and Hildebrant [1984].) This stochastic model requires average input values and provides average output values. In particular, for each part type, the input required is the average visit frequency to each machine (group) and the average processing time of an operation at each machine (group). The outputs include the steady-state mean production rates of each part type, machine utilizations, and average queue lengths at each machine (group).

MVAQ can also model, at an average, aggregate level, load and unload times, refixturing times, queueing times, and transportation times. The production ratios found by the methods described in §3 can be used to suggest numbers of pallets and fixtures of different types to maintain the calculated, optimal, aggregate ratios, as described in §3 and §4.1, say. These numbers can serve as input to the aggregate queueing network model. The output (machine utilizations) indicates how balanced the system is when the additional delay factors are included. In addition, the average production rates indicate if any due dates are in jeopardy.

Suri and Hildebrant [1984] indicate that this model is reasonably accurate and is about 10-20% pessimistic in its predictions, as compared to simulations of similar systems allowing more modeling detail. However, MVAQ is even more accurate in its relative predictions. For example, the ratios of the expected production rates of parts and machine utilizations matched those provided by simulation quite well. It is these relative values that would be of use here, to indicate ratios that provide a good balance.

Another useful model that could accept the production ratios found in §3 as input to help find the minimum inventory requirements is a timed Petri net. (See Dubois and Stecke [1983].) This model could complement the queueing network model because it uses deterministic operation times. Also, it is not an aggregate model. The actual processing times and part routes are modeled. Set-up times, transportation times, and queueing times are modeled in all detail, unlike the queueing network models.

For a certain subclass of timed Petri nets (in particular, decision-free nets), the graphical model can be easily translated into linear state equations in a {max, +}-based algebra. (See Cohen, Dubois, Quadrot, and Viot [1983].) Decision-free means that no decisions are to be made. Everything needs to be specified in advance, such as the part routes, the input sequence, and the like. A particular Petri net representation can be analyzed very quickly via some algorithms, based in part on Karp's [1978] efficient shortest path algorithm, to provide much information that is useful for performance evaluation.

Some of the output from the model includes the cycle time (hence the production rate), the bottleneck machine, its utilization, and the utilizations of

all other machines. Some particularly useful information specifies that production can be increased by either: adding a machine of a particular type; or inputting another pallet/fixture for a particular part type. We indicate how this information can be used via an example.

Prior to this, recall the following. In §3, we indicated how to find aggregate production ratios that balance the workload on all machines. In most of the many examples that were examined, including all of those discussed in §3 (except the example of §3.2.5 which we return to shortly), the aggregate ratios found also provided the minimum numbers of pallets and fixtures required. When the aggregate processing time information indicates an unbalanced machine workload (as the example of §3.2.5--see Table VI), then the minimum number of pallets required per part type can be much less than the specified ratios. We use this example of Table VI to demonstrate how the Petri net model and the information provided can be easily used to determine the minimum inventory requirements.

The information that the Petri net program requires is: i)for each part type, the aggregate production ratios (the a_i); ii) also for each part type, the number of pallets/fixtures dedicated to that part type; iii) a part input sequence.

For the example described in §3.2.5, this information is: i) $(a_1, a_2) = (1, 10)$; ii) $(n_1, n_2) = (1, 4)$; (This is just to demonstrate. We know via simulation that the minimum number of pallets required is: (1, 3)); iii) (1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2).

The output from the Petri net program would indicate that we have too many pallets/fixtures for part type 2. Changing (n_1, n_2) to (1, 3) in the next run provides the information that: production is maximized, the machines are balanced, there is no idle time, and there are minimum inventory and buffer requirements to maintain these optimal production ratios of (1, 10).

Notice that the production ratios are also used to help find the optimal, perhaps periodic in between periods of breakdown, input sequence. It is very useful to know the relative numbers of each part type prior to the actual sequencing. Of course, most situations will not be as simple as this example of §3.2.5.

Finally, note that simulation will be the most frequently used evaluator of the aggregate ratios.

5. FUTURE RESEARCH NEEDS

The aggregate production ratios provide guidelines to determine an appropriate part input sequence. However, further research is required in developing a more precise algorithm to find a good part input sequence. Some applicable work along these lines has been done by Hitz [1980] and Erschler, Lévêque, and Roubellat [1982]. However, these have been in flow shop situations, and did not allow several alternative routes.

Easier and more automatic generation of the production ratios and inventory requirements is needed. Also, implementation of secondary criteria for choosing ratios is needed for the situations in which multiple sets of ratios balance the workload. Artificial intelligence techniques may be able to help here.

ACKNOWLEDGEMENTS

Part of this work was done while the author was visiting at Centre d'Etudes et de Recherches de Toulouse. The author would like to thank Eric Bensana, Gérard Bel, Didier Canals, and J.-B. Cavaillé for helpful discussions. The author's work was supported in part by NSF Grant No. ECS 8406407.

REFERENCES

BERRADA, MOHAMMED and STECKE, KATHRYN E., "A Branch and Bound Approach for Machine Load Balancing in Flexible Manufacturing Systems", Management Science, 1986, forthcoming.

BUZACOTT, JOHN A. and SHANTHIKUMAR, J. GEORGE, "Models for Understanding Flexible Manufacturing Systems", AIIE Transactions, Vol. 12, No. 4, pp. 339-350 (December 1980).

CAVAILLÉ, JEAN-BERNARD and DUBOIS, DIDIER, "Heuristic Methods Based on Mean-Value Analysis for Flexible Manufacturing Systems Performance Evaluation", Proceedings of the 21st IEEE Conference on Decision and Control, Orlando, FL, pp. 1061-1065 (December 1982).

COHEN, GUY, DUBOIS, DIDIER, QUADRAT, JEAN P. and VOIT, M., "A Linear-System Theoretic View of Discrete-Event Systems", Proceedings of the 22nd IEEE Conference on Decision and Control, San Antonio, TX (December 14-16, 1983).

DUBOIS, DIDIER, "A Mathematical Model of a Flexible Manufacturing System with Limited In-Process Inventory", European Journal of Operational Research, Vol. 14, No. 1, pp. 66-78 (January 1983).

DUBOIS, DIDIER and STECKE, KATHRYN E., "Using Petri Nets to Represent Production Processes", Proceedings of the 22nd IEEE Conference on Decision and Control, San Antonio, TX (December 14-16, 1983).

ERSCHLER, JACQUES, LÉVÊQUE, DIDIER and ROUBELLAT, FRANCOIS, "Periodic Loading of Flexible Manufacturing Systems", Proceeding of the IFIP Congress, APMS, Bordeaux, France, pp. 327-339 (August 24-27, 1982).

HITZ, K. L., "Scheduling of Flexible Flow Shops-II", Report No. LIDS-R-1049, M.I.T., Cambridge, MA (October 1980).

KARP, RICHARD M., "A Characterization of the Minimum Cycle Mean in a Digraph", Discrete Mathematics, Vol. 23, pp. 309-311 (1978).

KUSIAK, ANDREW, "Loading Models in Flexible Manufacturing Systems", WP #05/83, Department of Industrial Engineering, Technical University of Nova Scotia, Canada (March 1983).

SCHRIBER, THOMAS J. and STECKE, KATHRYN E., "Machine Utilizations and Production Rates Achieved by Using Balanced Aggregate FMS Production Ratios in a Simulated Setting", Proceedings of the Second ORSA/TIMS Conference on Flexible Manufacturing Systems: Operations Research Models and Applications, Ann Arbor, MI, Elsevier Science Publishers B.V., Amsterdam, pp. 389-400 (August 12-15, 1986).

SHANTHIKUMAR, J. GEORGE and BUZACOTT, JOHN A., "Open Queueing Network Models of Dynamic Job Shops", Working Paper No. 79-024, Department of Industrial Engineering, University of Toronto, Canada (August 1979).

SHANTHIKUMAR, J. GEORGE and STECKE, KATHRYN E., "Reducing Work-in-Process Inventory in Certain Classes of Flexible Manufacturing Systems", European Journal of Operational Research, Vol. 26, No. 2, 1986, forthcoming.

SOLBERG, JAMES J., "A Mathematical Model of Computerized Manufacturing Systems", Proceedings of the 4th International Conference on Production Research, Tokyo, Japan (August 1977).

SOLBERG, JAMES J., "Analytical Performance Evaluation of Flexible Manufacturing Systems", Proceedings of the 18th IEEE Conference on Decision and Control, San Diego, CA, pp. 640-644 (December 1979).

STECKE, KATHRYN E., "Formulation and Solution of Nonlinear Integer Production Planning Problems for Flexible Manufacturing Systems", Management Science, Vol. 29, No. 3, pp. 273-288 (March 1983).

STECKE, KATHRYN E., "A Hierarchial Approach to Solving Machine Grouping and Loading Problems of Flexible Manufacturing Systems", European Journal of Operational Research, Vol. 24, No. 3, pp. 369-378 (March 1986).

STECKE, KATHRYN, E. and MORIN, THOMAS L., "The Optimality of Balancing Workloads in Certain Types of Flexible Manufacturing Systems", European Journal of Operational Research, Vol. 20, No. 1, pp. 68-82 (April 1985).

STECKE, KATHRYN E. and SOLBERG, JAMES J., "The Optimality of Unbalancing Both Workloads and Machine Group Sizes in Closed Queueing Networks of Multiserver Queues", Operations Research, Vol. 33, No. 4, pp. 882-910 (July-August 1985).

STECKE, KATHRYN E. and TALBOT, F. BRIAN, "Heuristic Loading Algorithms for Flexible Manufacturing Systems", Proceedings of the Seventh International Conference on Production Research, Windsor, Ontario, Canada (August 22-24, 1983).

SURI, RAJAN, "Robustness of Queueing Network Formulae", Journal of the Association for Computing Machinery, Vol. 30, No. 3, pp. 564-594 (July 1983).

SURI, RAJAN and HILDEBRANT, RICHARD R., "Modelling Flexible Manufacturing Systems Using Mean Value Analysis", Journal of Manufacturing Systems, Vol. 3, No. 1, pp. 27-38 (January 1984).

SURI, RAJAN and WHITNEY, CYNTHIA K., "Decision Support Requirements in Flexible Manufacturing", Journal of Manufacturing Systems, Vol. 3, No. 1, pp. 61-69 (January 1984).

WHITNEY, CYNTHIA K. and GAUL, THOMAS S., "Sequential Decision Procedures for Batching and Balancing in FMSs", Proceedings of the First ORSA/TIMS Conference on Flexible Manufacturing Systems: Operations Research Models and Applications, Ann Arbor, MI, pp. 243-248 (August 15-17, 1984).

YAO, DAVID D. W., "Some Properties of Throughput Function of Closed Networks of Queues", Technical Report, Columbia University, New York, NY (1984).

A KNOWLEDGE BASED SYSTEM FOR DYNAMIC MANUFACTURING REPLANNING

Virginio Chiodini

Honeywell Computer Sciences Center
1000 Boone Avenue North
Golden Valley, Minnesota 55427

Most existing requirements planning systems can be characterized as "automatic top-down, manual bottom-up", referring to the fact that their algorithms develop a material plan by starting with a top level master schedule, and exploding it level by level down through the product structure, to create the supporting time phased material plan for each element in the product. As perturbations occur at any level in the product structure, the plan must be revised.

A replan that still meets the master schedule requirements is preferred. The development of this alternate plan works from the product level where the perturbation occurred back up the product structure, until a point is reached where the new plan conforms to the original plan. If such a point is not found, a revision of the master schedule is executed.

The "bottom-up" schedule revision is made possible by the slack existing in the original plan.

Traditional "Manufacturing Resource Planning" systems introduce slack in the schedule mainly in the form of overestimated lot queue times (waiting to be processed) and move times (among subsequent operations). Furthermore, the scheduling logic of the MRP systems is normally based on the assumption of fixed batch sizes, and does not exploit the possibility of overlapping contiguous operations on the same batch. Slack is needed not only for adjusting the schedule in the event of perturbations, but mainly because MRP schedules do not fully account for the actual capacity, and therefore the plan may result in over- and underload condition on the shop floor.

As the factories approach "just in time" (JIT) objectives (continuous (repetitive) production has JIT characteristics), the slack in queue time and move time and the possibility of overlapping are reduced or eliminated:

- The work centers are normally loaded at a higher percentage of their capacity.
- Inventory of work-in-process is reduced to a minimum.

Thus the replanning function supporting such an environment is much more restricted in its alternatives. Problems occurring at one one level of the process structure very often cannot be bounded at the level in which they occurred, and propagate to the superior levels, up to the Master Schedule.

A revision of the Master Schedule that still meets the production objectives or minimizes the impacts on them, must be developed within the conflicting constraints imposed by resource and material availability of

non empty production lines, and must be consistent with quality and utilization guidelines.

In this environment an effective production control system must be supported by two main tools:

- o A real time information system, to monitor the status of the shop floor.

- o A dynamic rescheduling function, capable to react to any anomalous condition occurring in the production process. This function must be able to predict the consequences of the perturbation occurred, and develop a new schedule that minimizes the impact on the production objectives, and limit the schedule revision within the smallest possible area of the shop floor.

The dynamic revision of a factory schedule in most of the high-volume repetitive manufacturing environments (or environments approaching a "Just In Time" context) is currently a manual job, assigned to teams of highly trained schedulers. The stochastic nature of the disturbances, the large number of alternatives to be explored, the rapid response time required, and the complexity of the constraints, make the solution of the problem unfeasible for mathematical programming techniques. On the other hand, the manual solutions are often tedious and affected by oversights, especially when multiple anomalous events force the single scheduler or the joint scheduling team to revise the schedules frequently.

APPLICATION SCENARIO

In August 85 we started an investigation of the applicability of Artificial Intelligence technology to the " real-time rescheduling" of a manufacturing system. As a test case we selected a plant whose production environment is evolving toward a JIT context.

The plant can be classified as a "high volume" production facility. The three final assembly lines produce 12,500 final units per day.

The plant does not utilize MRP systems, as they are not considered adequate to the specific production requirements. The planning activities are driven by a computer generated "Main Schedule" which is also the dispatch list of the final assembly lines. The Main Schedule establishes on a daily basis the sequence in which the models of the final product are to be assembled at the final lines. For each model, the Main Schedule specifies the lot size and the manpower required. The Main Schedule is firmed for a period of three weeks.

On a weekly basis the temporal scope of the Main Schedule is advanced of one week. The schedule of the added week incorporates the production requirements that appear on a long range Master Plan, updated according to the actual orders.

The Main Schedule is updated every night to take into account the actual daily production, the current and forecasted resource availability of the plant and, sometimes, new "crash" orders. The supervisor of the Main Schedule coordinates the work of six schedulers that are assigned to six different scheduling areas.

A short-range version of the Main Schedule with a temporal scope of 5 days is distributed to the team of schedulers. This is dynamically updated during the day to reflect the changes that, from time to time, might become necessary.

Each scheduler manually compiles every day the initial schedule of the assigned area. These schedules are then dynamically revised according to the evolving status of the shop-floor.

A "Production Planning and Control System" (PP&C), that collects and elaborates data received from a set of sensors applied to the key points of the shop floor, is being installed.

The schedulers are continuously informed by PP&C about the status of the work centers. The "Production Planning and Control System" reports information like:

- The status of the processing systems.
- The allocation of the manpower.
- The quantities of parts stored in the buffer areas.
- The production rate at each process unit.
- The percentage of parts rejected at the inspection points.

Based on the incoming data, each scheduler has the responsibility to determine whether the current schedule is in jeopardy and, if it is, to revise it in order to meet the demands of the assembly lines. When the nature and the duration of the disturbance prevents the support line from meeting the requirements of the assembly lines, a real-time revision of the Main Schedule is performed.

The real-time revision of the Main Schedule is limited in temporal scope to two or three shifts, starting from the time in which the perturbation occurred. Schedule revisions that extend beyond this temporal scope are performed by the batch rescheduling process.

The real-time coordinated rescheduling usually involves only the support lines that, with a possible update of their specific schedules, may contribute to a successful revision of the Main Schedule. The other lines, even if they originated the perturbation that is causing the rescheduling, may only constrain the number of possible rescheduling alternatives, with their current and expected availability of components.

At the test-site plant, we identified a set of support lines that are usually involved in the coordinated rescheduling. These lines, that will be referenced in the next sections as "Active Support Lines", are:

- The "Mainframe Line".
- The "Flatware Line".
- The "Roundware Line".
- The "Plastics Line".

These four Active Lines and the three Final Assembly Lines will be the direct objective of the real-time rescheduling system, the other lines and the parts they process will just play the role of sources of perturbation events and constraints applicable in the real-time rescheduling process.

The process structure of the Active Lines is outlined in Fig. 1, and will be described in the next section.

THE PROCESS STRUCTURE

The Final Assembly Lines.

The final product is produced in approximately 500 models that are grouped into two major classes:

- o "B" - assembled by the final Assembly Lines 1 and 2.
- o "L" - assembled by the final Assembly Line 3.

From a scheduling standpoint the final models are classified according to the importance of their due date requirements. Some of the models must be shipped during the same day in which they exit from the assembly lines. This class includes all the models called "special", that require higher manpower and special processes.

For other models the goal is to satisfy a cumulative demand over a certain period of time. A third class includes models for which the requirement is to maintain an average production rate over a long period of time.

Lines 1 and 2 have identical throughput and process time. They both process models of class "B", although some special models of this class can be assembled only in one of the two lines. The assembly of these special models is constrained in time and lot size. Furthermore the periods in which these special models are processed at the two lines are mutually constrained by the need for sharing additional manpower.

Line 3 assembles only models of class "L". It has the same throughput as lines 1 and 2, but a longer process time.

Each line processes approximately 12 different models every day. The assembly lines run in two shifts.

The "Mainframe Line"

Mainframes undergo three operations:

- o Press and weld (4 systems).
- o Ground coat (1 system)
- o Paint (2 systems: System A, System B)

All the weld systems are dedicated to processing a unique model of the mainframe. The activity of two of the systems is constrained by the manpower and by the throughput of the conveyor line that moves the units to the next storage area. These two systems are not normally run at the same time. The throughput of another system is slightly lower than the demand of the assembly lines. For this reason it is run during three shifts.

Paint System A paints preferably mainframe units addressed to assembly lines 1 and 2.

Some of the colors can be painted only by a specific system, or are

PROCESS STRUCTURE OF THE "ACTIVE SUPPORT LINES" AREA

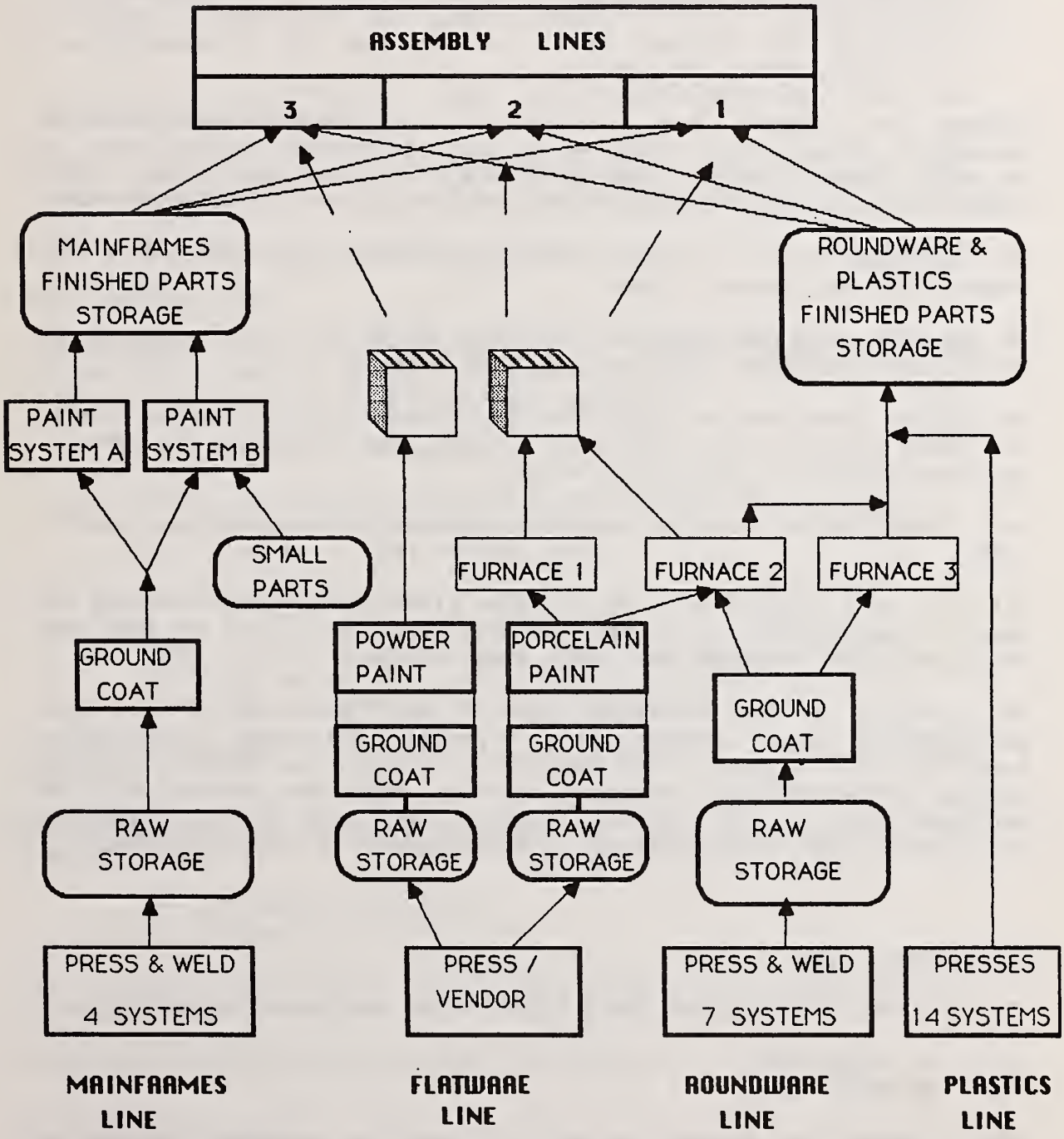


fig. 1

constrained in the period of time in which they can be painted, as they require additional manpower.

Paint System B is in charge of painting other special components that are fabricated outside the Mainframe Line. The painting of these parts has a lower priority, as they have low "Just In Time" requirements.

Paint System B is also used as a back up when the Powder Paint system of the Flatware Line is down. In this case Paint System B runs for three shifts.

Two storage areas exist:

- o One for raw units, after the weld operation (Raw Storage).
- o Three for painted units, at the head of the assembly lines. (Paint Storage).

Buffers are utilized just to decouple the different throughputs of two subsequent process units. They may be used as temporary storage areas, up to their capacity, only in case of failure of the next process unit. This enables the work center to recover the lost production by using overtime.

The mainframe units are moved through the various operations and to the assembly line by conveyor lines:

The Raw Line moves the mainframe units from the press and weld area, to the Raw Storage, the Ground Coat, and the Paint Systems.

The Paint Lines move the units from the intersection with the Raw Line to the Paint Systems, and to the Paint Storage. The Paint Lines may feed all the three sections of the Paint Storage.

The Feeder Lines move the units from the Paint Storage to the assembly lines.

All the work centers of the Mainframe Line normally operate during two shifts, synchronously with the assembly lines, except for the weld unit mentioned above, that operates during three shifts.

The schedule of the Mainframe Line is very sensitive even to minor perturbations (i.e. abnormal scrap of parts). The number of constraints that must be satisfied, the complexity of the process dynamics, and the reduced work-in-process inventory allowed, make the scheduling of the Mainframe Line a very complex function, and require rapid schedule revisions in case of malfunctions, to avoid impacts on the Main Schedule.

The Flatware Line

The routing of the Flatware Parts includes two contiguous operations:

- o ground coat
- o paint

The Flatware Line processes two parts for the Final Assembly Lines and two other parts that are directly shipped to another plant. The colors painted match those applied to the mainframes.

The Flatware Line includes two systems that perform both operations:

- o The Powder Paint System
- o The Porcelain Paint System

Each paint system may apply the whole set of colors on all the parts. The components painted at the two systems are not interchangeable in the final models. The two paint systems have different throughput and process times. One of the process unit of the Porcelain Paint System is shared with the Roundware Line.

The Porcelain Paint System is also used to process the parts that are directly shipped to a different plant. The process of these parts normally has higher priority over the Flatware Parts directed to the final assembly lines.

The finished components are moved from the Flatware Line to the final assembly lines in containers.

Most of the Flatware raw parts are fabricated in a different plant, the others are fabricated in the press department of the same plant.

Both Paint systems run during three shifts.

The Roundware Line

The Roundware Line processes two parts for the local assembly lines and another part that is shipped to a different plant. The two parts that enter into the assembly lines undergo two operations:

- o Press and weld (7 systems)

  Half of the press and weld systems are dedicated to process a unique part and type, the others are dedicated to a single part, but may process multiple types.

- o Ground coat (2 systems)

  One of the systems (shared with the Flatware Line) is normally dedicated to process one part, the other processes both parts addressed to the assembly lines.

The parts shipped to a different plant undergo only to the Press & Weld operation.

The Roundware Line has two storage areas for the parts entering into the assembly lines:

- o The Raw-Parts Storage area.
- o The Finished-Parts Storage area.

The Roundware units flow through all the operations and storage areas via conveyor lines.

The Roundware Line run during all the three shifts.

The Plastics Line

The Plastics Line processes a large number of parts. Some of them enter

directly into the assembly lines, others are subcomponents of other parts. Only the processes of two of these parts, that enter directly into the final assembly line have been included in the Active Support Lines area. Both parts are processed with a single operation.

The first of these parts is processed by 9 systems, 2 of which are dedicated to a unique type. The units of this component are moved in containers to the assembly line.

The other component is processed by 5 systems, 1 of which is dedicated to a unique type. The units of this component are moved to the same final storage area used by the Roundware Line and share the last portion of the same conveyor line.

The Plastics Line runs during all the three shifts.

THE FUNCTIONS OF SCORE

SCORE (Shop-floor COntingency Rescheduling Expert) is a prototype knowledge based system for real-time rescheduling of the shop floor, currently being developed at the Honeywell Computer Sciences Center.

The objective of the prototype is to demonstrate the applicability of A.I./E.S. technology to the real-time rescheduling of the manufacturing environment described in the previous section.

Although the whole shop floor is considered as a potential source of perturbations that propagate to the final assembly lines, SCORE will limit its rescheduling scope to the "Active Support Lines" area.

During the visit at the plant, we realized that, in this environment, a real-time rescheduling system can effectively reach its objectives if it is able to provide the following integrated set of capabilities:

- Schedule the Active Support Lines, on the basis on the Main Schedule of the final assembly lines.
- Predict the propagation of abnormal events occurring at the Active Support Lines.
- Reschedule each Active Support Line, when abnormal events occur inside the line itself, or when the Main Schedule of the final assembly lines is being revised.
- Coordinate the schedule revision of the final assembly lines and of the Active Support Lines, in consequence of abnormal events occurring on the shop-floor.

The function of generation and long-range revision of the Main Schedule will still be supplied by the current batch scheduling system utilized at the plant.

THE SUPPORT LINES SCHEDULING FUNCTION

The scheduling function of the Active Support Lines has been included in SCORE for its tight connection with the logic used in rescheduling the same lines.

The schedules of all the Active Support Lines are "pulled" by the Main Schedule of the final assembly lines. Each support line has different JIT requirements (level of work-in-process inventory allowed) that depend on its characteristics. Inside a support line, the level of work-in-process inventory is different at different points along the routing of operations. Usually the JIT requirements are more restrictive at the end of a line.

SCORE will not address the problem of reducing the work in process inventory, and will assume the constraints that are currently accepted at the plant.

The main problem that the scheduling function has to solve is how to dispatch and split the lots at the process units of a work center.

Each work center may be represented as a set of parallel process units. Normally a specific type of component may be processed at more than one unit. Each process unit has different processing characteristics (set-up time, throughput, move time to the next storage area or work center).

The scheduling function has to define, for each lot of components required at the final assembly lines, the following entities:

- Which process unit should be used.
- How many times the lot should be split to enable the process units to satisfy the pending demands from all the requestors, without violating the constraints imposed by the structure and dynamics of the system, and by the operational procedures.
- The start-processing time and the size of each sublot.

The approach used by the schedulers to solve this problem is mainly a constraint directed reasoning. First are scheduled the lots with a small number of scheduling alternatives (for example small lots that "must" be assigned to a specific processing unit). They define "anchor points" that partition the scheduling interval into subintervals. Then the other lots are scheduled according to their urgency. The interaction among the requirements of the different lots, and the processing characteristics, are utilized to define dynamically further constraints that normally lead the problem to a solution with a minimum amount of backtracking.

The method of problem solving adopted in SCORE is similar to the constraint directed reasoning approach to scheduling developed in ISIS [FOX83].

When the routing of a component includes more than one operation, the schedules of the work centers that perform these operations must be coordinated. The approach taken is simply to schedule the most critical work center first, and then use this schedule as a constraint applicable in scheduling the other work centers.

The criticality of a work center mainly depends on two characteristics:

- The limited hedge of throughput available against the demand of the assembly lines.

- The high set-up time required for model change.

The demands of the assembly lines are propagated backward in time, using a critical path method, and the earliest start time and latest finish time of each demand at the critical work center are evaluated. A feasible schedule is then searched using the constraint directed approach described above.

THE RESCHEDULING FUNCTION

When an unexpected abnormal event occurs on any one of the support or assembly lines, the goal is to eliminate or minimize the effects on the overall production plan by executing the following steps:

- Detect and identify the problem.
- Determine the effect of the perturbation on the schedule of the line where the event occurred, and evaluate the impact on the capacity of the line to satisfy the demands.
- Revise the schedule of the perturbed line in order to eliminate any impacts on the other lines.
- Perform a schedule revision at the superior level of the process structure, if the problem cannot be restricted within the perturbed line.

The logic flow of the rescheduling function is shown in Fig. 2.

Classification of the perturbation events.

The scheduling function receives a great support from the continuous, real-time information on the status of the shop-floor, supplied by the Production Planning and Control system. However the continuous flow of information requires a deep knowledge of the dynamics of the production process, to be synthesized. Incoming data need to be timely monitored, and evaluated in comparison with the current schedule to convey significant warnings.

A real-time rescheduling decision-aid system should be able to monitor the incoming data at crucial points in time, and convert quantitative into qualitative information.

Prediction of the events propagation.

The level of perturbation that the abnormal event is going to introduce into the current schedule must be inferred from the type and the duration of the event, the dynamics of the production process, the current status and the schedule of the shop floor. The evaluation of the effects of the perturbations can immediately focus the attention on the most promising corrective actions, avoiding the exploration of alternatives with low probability of success.

For example an abnormal percentage of scrapped parts might be immediately corrected by rework if:

- There is slack before the process of the subsequent lot.

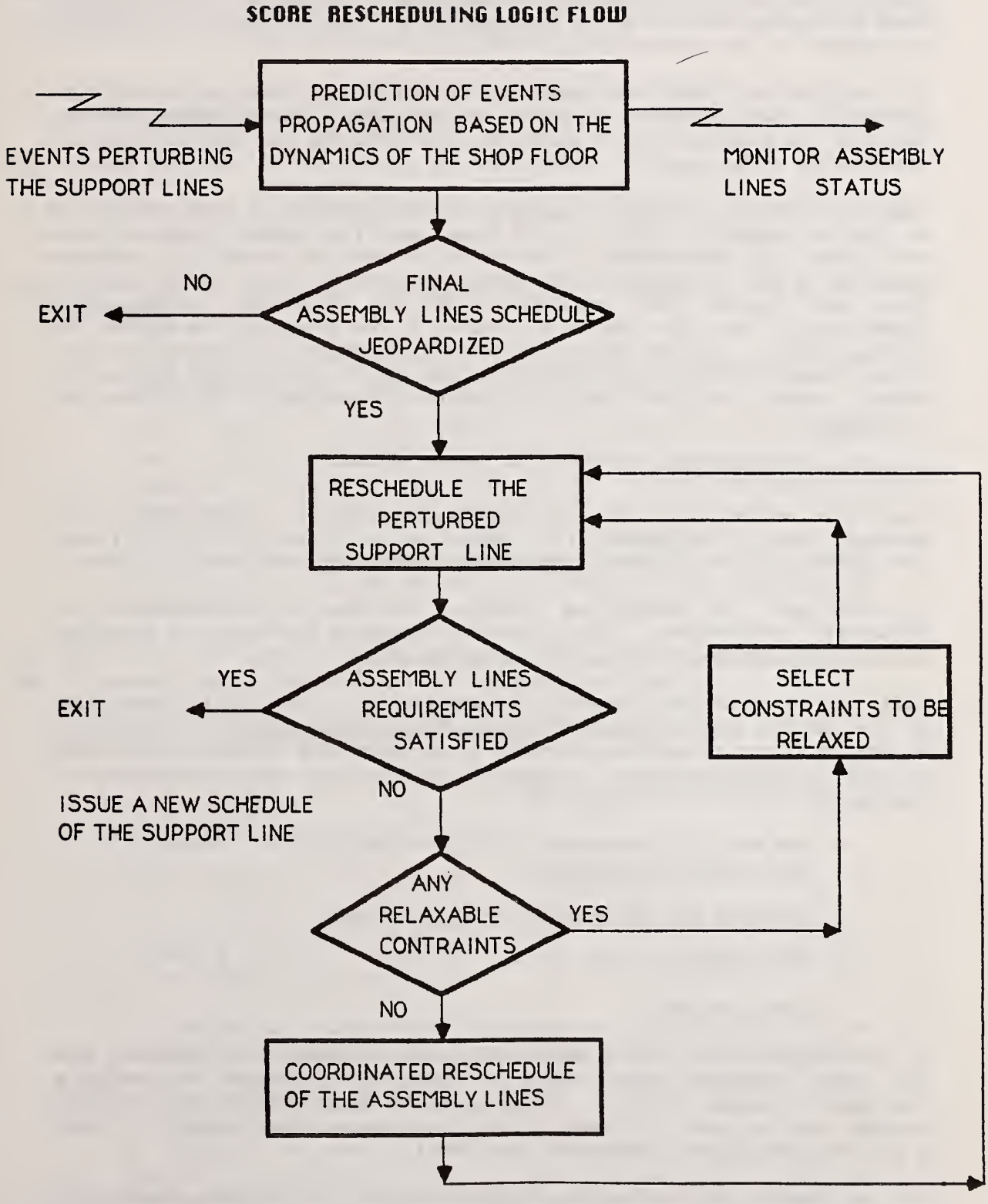

SCORE RESCHEDULING LOGIC FLOW
PREDICTION OF EVENTS PROPAGATION BASED ON THE DYNAMICS OF THE SHOP FLOOR
EVENTS PERTURBING THE SUPPORT LINES
MONITOR ASSEMBLY LINES STATUS
NO
EXIT
FINAL ASSEMBLY LINES SCHEDULE JEOPARDIZED
YES
RESCHEDULE THE PERTURBED SUPPORT LINE
YES
EXIT
ASSEMBLY LINES REQUIREMENTS SATISFIED
SELECT CONSTRAINTS TO BE RELAXED
NO
ISSUE A NEW SCHEDULE OF THE SUPPORT LINE
ANY RELAXABLE CONTRAINTS
YES
NO
COORDINATED RESCHEDULE OF THE ASSEMBLY LINES

fig. 2

- o The dynamics of the line allows an immediate rework of the scrapped units or there is a set of compatible units stored near to the process unit.

When both of these conditions are satisfied, the problem is normally handled by the operators according to the standard procedures, and no schedule revision is requested. Historical information can anyway help in determining whether the problem is temporary or persistent. In the latter case a schedule revision might be suggested to compensate for the decreased throughput of the process unit.

If the problem cannot be handled by the normal operating procedures, the prediction function should determine the probability for a limited schedule revision to eliminate the problem, without affecting the superior level of the process structure.

When the work-in-process storage area located between two work centers has a limited capacity (this is the normal case for support lines operating with high JIT requirements), the effects of abnormal events may propagate backward as well as forward. For example a major slow-down of the assembly line may cause the congestion of the downstream storage area. Consequently, the jam may be propagated to the conveyor line feeding the storage area. If the conveyor line is also feeding the storage areas of other assembly lines, the congestion may cause a generalized shortage of parts, unless the downstream work centers are rescheduled to overcome the contingency.

Local schedule revision.

When the satisfaction of the assembly line demands is jeopardized by an abnormal event, the general rule applied at the support line is to revise the schedule, allocating all the available resources to solve the problem.

In this case the scheduling function must face an increased number of structural constraints, like a possible decreased throughput of a process unit and the non-empty status of the input conveyor lines.

Most of the optimization constraints applied during the initial generation of the schedule are considered "relaxable" in this situation. Optimization constraints are classified according to an importance hierarchy determined by their cost effectiveness. A possible list of constraint relaxations, in increasing order of importance, is:

- o Use the full throughput of the processing unit. (Instead of the normal throughput).
- o Increase the inventory of work-in-process.
- o Use overtime.
- o Add a new shift.

A pre-rescheduling analysis must determine what constraints, starting from the least important ones, must be immediately relaxed, to enable a successful schedule revision. The constraints above this threshold will be relaxed only in case of failure of the first rescheduling attempt, to allow a second search to be potentially successful.

If the search for a feasible reschedule fails, the rescheduling function must determine when and how the Main Schedule is going to be affected.

Coordinated schedule revision.

When the effects of a perturbation event cannot be limited to the single support line, the Main Schedule must be revised in order to avoid or minimize losses of production.

The scheduling function tries to change sequences, quantities, and models of the lots scheduled at the assembly lines, to reach the overall production objectives within the constraints imposed by the reduced availability of parts. The production objectives may be classified according to the following top-down hierarchy:

1) Process the lots of "Special Models" and the lots of models that are highly constrained in due time.

2) Avoid any loss of production at the assembly lines.

3) Process the lots of models for which a specific cumulative demand exists over a period of time.

The revision of the Main Schedule is made possible by three main circumstances:

- o The partial commonalities of components existing among the final models that appear in the Main Schedule.

- o The different JIT requirements of the feeding lines. (This may enable an assembly line to anticipate the process of a specific model if the prior model is in shortage of a component).

- o A timely warning for a possible shortage of parts, issued either by the prediction or by the support line rescheduling function.

The revision of the Main Schedule requires the coordination of a complex set of assumptions, propositions, and constraints like:

- o The processing constraints of the assembly lines.

- o The urgency of the lots that appear in the Main Schedule.

- o The process and product structure.

- o The expected and the current availability of parts.

- o The capability of the support lines to meet the requests of changing their schedule. This specific knowledge is structured at two levels:

  A synthetic, high level visibility of the dynamics of the manufacturing system, that enables the rescheduling function to focus on the most promising alternatives that are likely to be supported by feeding lines.

  A detailed visibility of the dynamics, constraints and status of a support line. This ultimately verifies the feasibility of a rescheduling request, when the result of

the application of the higher level knowledge is uncertain. This evaluation may imply the execution of an actual partial reschedule, and may be computational expensive.

The revision of the Main Schedule is furthermore constrained by the requirement of a rapid generation of the new schedule. For this reason the coordinated rescheduling function must search for solutions that avoid or minimize computationally expensive sequences, like the requests of changing the schedules of the support lines. The search process must therefore be focused on rescheduling alternatives that:

- o Are closer to the current Main Schedule.
- o Do not alter the cumulative demand of a specific component.
- o Limit the number of support lines involved in the rescheduling process.
- o Limit the interval of time affected by the revision.
- o Limit the number of assembly lines to be rescheduled.

When a schedule revision of the support lines cannot be avoided, the knowledge of the product and process structure, and of the current and expected status of the work centers, is used to direct the attention on the support lines that are the most promising candidates to be rescheduled.

A support line becomes a candidate to be rescheduled when two conditions exist:

- o A shortage in a specific type of component, that the line produces, is precluding a promising solution during a particular time interval.
- o The request to modify the schedule, to make the requested component available, is "timely" for the time interval in which the component is needed.

The possibility for a request to be timely, depends on a complex combination of elements like:

- o The lead time of the component required.
- o The possibility of preemption of the current lot being processed at the work center.
- o The status and the resources available at the work center.

A component may be described as a combination of attributes. A specific combination determines the "type" of a component. Each attribute is bound to a specific value by an operation. As soon as an operation becomes "timely" for a schedule change, the attribute defined by the operation may assume a larger set of values.

The time-phased throughput of a feeding line is therefore bound to the types and quantities of components currently scheduled, until the last operation (the nearest to the assembly line) becomes "timely" for a schedule change. Moving ahead in time, more schedule-change requests become "timely", and the number of potential alternatives for revising the Main Schedule increase.

The revision the Main Schedule is composed of a set of tasks. Each of these is in charge of accomplishing one of the production objectives assigned to the revision process, and is dispatched according to the priority of the goal. Each task explores the various alternatives available to satisfy its objective, within the space of the time phased availability (or potential availability) of components, and within the constraints imposed by the process structure of the assembly lines. The resources available (components, process time, manpower, etc) are therefore "reserved" by a task to an objective according to its priority.

When multiple alternatives exist to the attainment of a goal, they are ranked according to a heuristic evaluation of the probability that the partial solution will rapidly lead to an acceptable global solution.

The rating of the partial solutions tries to focus the search process on paths that avoid or minimize the deviations from the current schedule of the shop floor.

The highest priority task has the objective of satisfying the demand of "special models".

This is performed by identifying the periods in which the special models can be processed, within the constraints of the assembly lines, and the availability of components from the feeding lines.

When the first step ends, either reporting full success, partial success or complete failure, the second goal (avoid or minimize the process interruptions of the assembly lines) assumes the highest priority. This second task proceeds forward from the time in which a "shortage warning" has been issued by the prediction function. This decision is based on the assumption that the number of solutions available during the first period, until more components become available, or more schedule changes become timely, is very restricted.

Actually, during this interval, the availability of components is limited to the quantities and types that have been scheduled according to JIT constraints. Furthermore, a portion of these quantities might have been already "reserved" by the previous task. The restriction in components availability must be compensated by extending the exploration to all the models of the final product that can match the set of components available. Any models, even those located late in the five-days Main Schedule, are candidate to be processed during this period.

The alternative combined solutions generated by the first two tasks, are then evaluated by the third task, according to their capabilities to satisfy particular time-constrained cumulative demands of final models.

The revision of the Main Schedule proceeds forward in time until one of the following two conditions is met:

- o A point is found where the revised schedule matches the current schedule, without affecting the cumulative demand of components of the subsequent lots of final models.

- o The schedule revision has been extended to a point in time in which subsequent revisions may be adjusted by the batch rescheduling functions.

The coordinated rescheduling of the assembly lines is ultimately verified

by the local schedule revision of the support lines.

THE DEVELOPMENT OF SCORE

SCORE is being developed on a Symbolics 3600, using Carnegie Group's Knowledge Craft as a software building tool.

The initial prototype will be logically integrated with the Production Planning and Control System. The real-time information received from the "Production Planning & Control System" will be simulated.

SCORE requires the access to data (for example the product structure) that are stored in the central database. The current approach consists in replicating the required (low-volatile) data in the knowledge base.

REFERENCES

[FOX83] M. S.. Fox - "Constraint-directed Search: A Case Study of Job Shop Scheduling."
PhD thesis Carnegie-Mellon University, 1983.

DISPATCHING - THE CRITICAL AUTOMATION LINK

by Henry A. Watts

PROMIS Systems Corporation
4699 Old Ironsides Drive
Suite 300
Santa Clara, CA 94086

THE IMPORTANCE OF DISPATCHING IN AUTOMATED MANUFACTURING FACILITIES

he PROMIS CIM system is now in use at semiconductor manufacturing acilites as a production tool, so this paper reflects much more the inking of an applications-oriented manager than that of a theorist. evertheless, I hope to shed some light on what we have learned about nking overall production planning to the dispatching function, specially with respect to the challenges posed by integrated quipment automation manufacturing in a complex environment.

he following factors must be taken into account by any production cheduling system that expects to stand up to serious use in the emiconductor wafer fabrication manufacturing environment:

* The process often will use the same type of machinery several times, making it impossible to use standard, straight-through models of processing.

* Certain steps in the process must be initiated within short and specified time intervals after the completion of the previous process step. Normally the prior step is a special cleaning of the material and the next step is a process sensitive to moisture or other types of contamination.

* It is imperative to assess the quality of work being done by certain pieces of processing equipment. A single wafer may be handled by equipment up to 150 times before wafer fabrication processing is complete, providing many opportunities for mis-processing, and a single piece of such equipment may handle up to $2,500,000 worth of work-in-process in a given week. This is equivalent to the total value of all of the production of the wafer fabrication area each week.

* The quality of the work being done by a machine is often not truly assessable until many steps later in a process where it is possible to take precise measurements.

* In an automated environment there are practical limits on the amount of work-in-process that can be staged in front of a given piece of processing equipment.

Dispatching, or choosing the specific lots of material to work on of the lots that are available to be worked on, may seem to be a mundane topic, compared with linear optimization programs and other intellectually challenging concepts. Nonetheless, dispatching can critically affect the challenges listed above. First, in any facility that uses one piece of equipment for more than one purpose, or at more than one point in the process flow, choosing the wrong lots can lead to drying up the supply of material at one of the workstations which follow, while overloading others. In general terms, a balanced line is defined as a manufacturing line that has the work-in-process evenly distributed so that all critical pieces of equipment can be kept busy and material must not wait unduly long before being processed. A line out of balance leads to reduced productivity. The notion of line balance, and the special problems created by the repetitive use of a single workcenter will be treated more thoroughly later. If the issue of line balance is not addressed in some constructive way the facility will poorly utilize the existing investment in capital equipment and other resources and thereby reduce its ability to compete in the marketplace.

In some cases the response to finding that critical equipment is sometimes idle due to lack of material is to keep more material active in the line, thereby reducing the probability of running out of material at any one workstation. One negative effect of this overloading of the production line is that the capital cost for the extra material in the line tends to reduce the economic competitiveness of the manufacturing facility. Another negative effect of overloading the line is that when quality inspections indicate that some machine is producing bad material, there will be more material between the guilty machine and the inspection point, resulting in more material being scrapped. This will be true in any manufacturing plant in which the effects of the work being done by a machine are not directly measurable at the machine, but must await further processing. It is especially true in semiconductor wafer fabrication.

The notion of dispatching correctly to avoid overloading a given operation capacity exists in much of the work world. Aircraft dispatching, for example, must carefully attend to the routine maintenance requirements of the aircraft, lest the entire fleet end up some morning parked in front of the maintenance hangar, every plane in need of some sort of maintenance, to the great dismay of the chief mechanic. The goal in this case would be to keep the entire fleet evenly distributed throughout the complete and total maintenance cycle so as to have a steady and predictable flow of work for the maintenance mechanics, as well as a steady availability of airplanes. Note that this brings us to a serious requirement of the exact maintenance status being a key determinant of what planes fly which routes, all in the interests of keeping the fleet 'maintenance balanced'. This problem is very similar to the general wafer fabrication problem in the semiconductor industry, as it involves the repeated use of a capacity by a process flow, the 'process' in this case being the lifetime of the airplane.

Integrated equipment automation brings special challenges in dispatching, because there will be less human judgment in the system. In the current rather manual wafer fabrication environment one may, at best, hand to the production operator an ordered list of the lots of material to be worked next. The operator may or may not deviate from this list for reasons that may or may not be reasonable. One of the great advantages and an equally great disadvantage of true factory automation is that the machinery will try to do just what it is told. The lack of human judgment in the specific decisions that are being made on a minute by minute basis indicates that the models and decision structures used must be quite precise. The lack of dispatching algorithms that keep the production line in balance in a non-automated environment is regrettable and inefficient; the same omission in an automated manufacturing environment is probably not survivable.

In the semiconductor industry the emerging transformation from rather manual processing to an intensive degree of automation presents an additional requirement to manufacturing control systems such as dispatching: since the change from manual to automated processing is a gradual change, the control systems need to work well in both environments. Where this is not true the user of such systems is presented with an 'automation cliff', a point up to which conventional methods work, and beyond which totally new methods are required. Such cliffs exist in real life, but they are to be avoided where possible. The jump off the cliff often presents an unacceptable danger to the continuous output of the production line.

AN OVERVIEW OF THE SEMICONDUCTOR INDUSTRY

To deal with these complexities a brief overview of the manufacturing requirements of this one industry may be useful. It is not the intent of this overview to cover all of the variations that have been used in semiconductor manufacturing, are in use now, or are anticipated. What is presented is a simplified view with the intention of acquainting the reader with the fundamental manufacturing and scheduling issues.

There is a general misconception, shared by many people in the semiconductor business, that semiconductor manufacturing is classifiable as light electronics. Semiconductor manufacturing is actually heavy applied chemistry. The nature of this chemical processing is not a linear sequence of equipment, but rather is characterized by repeated use of equipment, in some cases under very rigid time constraints. In the wafer fabrication area, thin circular wafers of a single silicon crystal (about .02" thick, 2" to 6" in diameter) are processed by repeated cycles of applying microscopic images, then inserting chemicals in the images, or of applying a coating of material such as aluminum for 'wiring up' the circuit, and, as the next step, making a pattern in the aluminum. In a typical process the machine that deposits the aluminum might be used only once or twice for a given wafer, but the work area that does the precise imaging, transferring patterns onto the wafers, will be used many

times. The actual number of uses is process-specific, but 10 cycles through the imaging or photo-masking area is typical, with numbers between 4 and 17 being common (see Figure 6). The result of the wafer fabrication activity is a wafer with anywhere from 40 to 50,000 or so individual circuits on it.

The steps that precede the wafer fabrication activity are product design and the creation of a set of photomask images, one image for each layer of processing to be done. The steps that come after wafer fabrication are electrically testing each circuit, cutting the wafer into individual circuits, putting each good circuit into a protective package and testing the packaged part. Each of these main phases of semiconductor manufacturing may happen in a different production area. Production areas in this sequence may be anywhere in the world.

The semiconductor industry can be characterized by fast technological change, an extreme level of competitiveness, very high capital equipment costs, with much of the production being equipment-dependent, long processes (involving as many as 400 specific processing steps in the wafer fabrication alone) that return several times to the same workcenter, processes that are very yield sensitive. In short, it is a production line very much in need of maintaining a balanced profile of work-in-process. The industry is also characterized by highly interactive technical factors, with a strong need for information feedback and feedforward, and a high need for cleanliness in the manufacturing area. Note that the net effect of the feedback is often a process change. In clear distinction to many other industries, many semiconductor manufacturing areas undergo process changes on a daily basis. Note also that the complex effects of changes in demand from the outside world must be rapidly reflected in the operating schedules of the manufacturing facility.

DISPATCHING MUST TIE TO A REAL DEMAND

In order to establish the measurement criteria for the dispatch list a higher-level view of the requirements of the production facility is needed. Whatever else the dispatch list does, it must direct the production facility to build material for which a demand actually exists.

The dispatch list required is one that would, if followed exactly, lead the production line into balance, and produce either exactly the parts that are needed as quickly as possible, or use the existing manufacturing capability to its fullest extent. This requires a definition of the term 'exactly the parts that are needed'.

A definition of the exact parts needed and the exact work therefore required of the manufacturing facility requires a link to a corporate-wide planning tool. This corporate-wide planner will use a list of parts needed, the dates on which they are needed (this need may be due to firm customer orders or may be internal estimates of future demand), known processing cycle times and product yields through the processing steps (cycle times to include shipping time

when the production areas are geographically separated), known processing capacities (equipment or labor limited) to produce a relatively feasible list of parts required out of each of the production areas and the date each batch of parts is needed. To the extent that one type of device early in the process may be used to make several different devices later in the process, this also must be taken into account by the planning system. Further, to the extent that such a planning system needs to anticipate material being started into a production facility, it must not exceed reasonable planned levels of starts absorption for that facility. The actual starts rate must be determined by the next lower planning module, the actual factory floor scheduling and dispatching system, in light of actual or forecasted on hand inventory, current capacity and recent yield and cycle time trends.

In addition to specifying the required output of each production area, the planning system must produce material requirements, time-phased supply and demand of each device, and should flag firm orders that are likely to become delinquent as well as work-in-process for which there no longer exists a demand (this could be due to order cancellation or unexpectedly high yields). This discussion on dispatching presumes such a corporate-wide planning capability.

DISPATCHING MUST HELP BALANCE THE LINE AND MEET THE REQUIRED OUTS

Once the actual demand-driven requirements for output from a specific production area are known, new considerations must be accomodated. What is needed is a schedule that, if exactly met, would lead the production line into balance, producing exactly the parts that are needed in the shortest time possible. In the case where demand exceeds production capacity the existing manufacturing capability must be run to its fullest extent, producing as nearly as possible the correct mix of product. As a practical matter the schedule must span at least a week of production (in the semiconductor industry), must show required activity at least down to the resolution of an 8-hour production shift, must show activity required by each operation in the process, and must group this required activity by the type of equipment used for the operation. To perform in this way the system must certainly have imbedded in it a concept of categories of equipment and must understand the common and differing ways in which each process uses each type of equipment.

Constraints on this plan will be available labor, available machine capacity (units per hour x working hours available x % uptime), known production yields, target cycle times and the existing distribution of the work-in-process. The distribution of the work-in-process is important, as machine and labor capacity needed to meet the output schedule may go unused if, at various points during the week, no work is available for the machine. Furthermore, a good scheduling procedure must predict the extent to which this will happen, or the schedule produced will not be feasible.

Target cycle times are a key system constraint since these times will be turned into a required level of work-in-process for each equipment type at each operation in the process. As long as the dispatching system works toward this and the machinery is available for the percentage of working hours that was assumed in the schedule, the achievement of the correct level of work-in-process and predicted throughput rate will drive the cycle time to its target level.

The procedures that such a local planning system must follow are relatively straightforward to describe, if not always quite so clean to implement in the complexity of the semiconductor manufacturing and product structure environment.

1. Check whether there is sufficient capacity to run the required production volumes on a steady-state basis. If not, then the stated demand on the production area is excessive and must be revised to an achievable, steady-state level. Typically this will be done by looking, say, 12 weeks out in the demand statement for a given process and calculating the average outs per week required for this process. This is the number to capacity check as a first order test of the feasibility of the schedule.

2. For each operation in the process calculate the required activity for the week. This is done by working from the end of the line backward toward the beginning of the line. If s is the number of steps in the process, A_n represents the activity or number of units to be processed at any step n, and Y_n is the expected yield at any step n, then

A_s = required outs/Y_s,

and, for any step n,

$A_n = A_{n+1}/Y_n$

n, Operation	1	2	3	4	5
Y_n, Predicted Yield	90%	95%	90%	98%	90%
A_n, Required Activity (steady state)	295	265	252	227	222
A_{n+1}, Required outs of n	265	252	227	222	200

Figure 1

3. Given that the equipment type and required labor for each operation are known, the entire set of processes can be summed by operation and by equipment type, and the percent utilization of both equipment and labor that is implied by the required outs schedule can then be determined. If this required utilization is too high, then either the outs requirement must be restated, or the computer can ratio down the entire outs requirement by (available capacity/required capacity). At this point an achievable steady-state production plan is known. There may or may not currently exist available capacity also to bring the line into balance.

4. Within each process the target cycle time can be transformed into a target work-in-process by the following formulas:

Reversed cum yield at operation n is defined to be the percentage of material at operation n that is expected (at predicted yield) to complete processing successfully through all s operations; if Y is yield, n is any step and there are s steps, then reverse cum yield is

$$RCY_n = Y_n \times Y_{n+1} \times Y_{n+2} \times \cdot \cdot \cdot \times Y_s$$

The quantity of work-in-process required, without correcting for yield (i.e., expressed in equivalent good units out(EFG)), where CT is the cycle time of any given operation, and where the time units are the same, is given by

$$TWIP(EFG)_n = CT_n \times \text{good units required out/time}$$

This can now be translated to the actual quantity of work-in-process required at operation n by

$$TWIP_n = TWIP(EFG)_n \; / \; RCY_n$$

n, Operation	1	2	3	4	5
Y_n, Predicted Yield	90%	95%	90%	98%	90%
A_n, Required Activity (steady state)	295	265	252	227	222
A_{n+1}, Required outs of n	265	252	227	222	200
CT_n, cycle time (days)	5	2	3	5	1
CT_n, cycle time (weeks)	1	.4	.6	1	.2
$TWIP(EFG)_n$	200	80	120	200	40
RCY_n	.68	.75	.79	.88	.90
TWIP, target work-in-process	295	107	152	225	44

Figure 2

5. Given that the steady-state runrates have been reality-checked and that the target inventories required at each operation are known, the process now works backwards through each process, creating a schedule that will bring the line into proper balance. Modifying the formula above to account for actual work-in-process at step n, WIP_n, and target work-in-process at step n, $TWIP_n$, the activity required, AR is given as,

$$AR = (AR_{n+1} + TWIP_{n-1} - WIP_{n+1}) / Y_n$$

n, Operation	1	2	3	4	5
Y_n, Predicted Yield	90%	95%	90%	98%	90%
A_n, Required Activity (steady state)	259	265	252	227	222
A_{n+1}, Required outs of n	265	252	227	222	200
CT_n, cycle time (days)	5	2	3	5	1
CT_n, cycle time (weeks)	1	.4	.6	1	.2
$TWIP(EFG)_n$	200	80	120	200	40
RCY_n	.68	.75	.79	.88	.90
TWIP, target work-in-process	295	107	152	225	44
WIP, actual work-in-process	2020	50	100	300	5
AR_n, Required Activity (line balancing)	372	278	213	266	222

Figure 3

The calculation is not so simple, however, because there may be insufficient inventory available to operation n to meet this level of activity. The actual computer processing needs to determine, for each step n, before working any further backwards to n-1, how how much inventory will be available to operation n, that is, how much inventory actually sits within cycle-time reach of operation n. Obviously, this material must be yielded from its current location to the location at which it will be needed, or the estimates of available material will be too optimistic. The simplest operating presumption seems to be that material is evenly distributed within the processing step where it sits. This obviously is subject to strong improvement in situations where the tracking system can discriminate between material in process (knowing the time when processing started), and material awaiting processing.

The view of the material available would, presuming an even distribution within its current operation, look as follows:

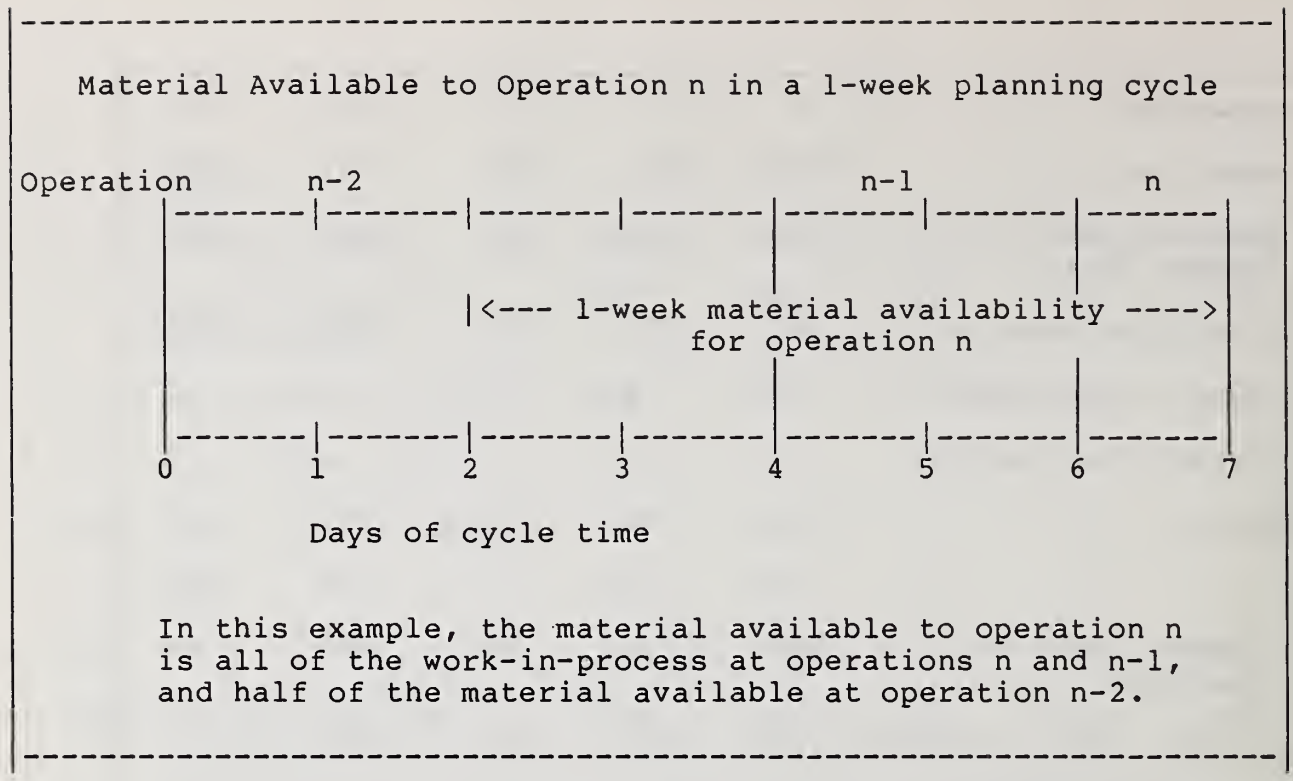


Figure 4

[With sufficient computing power, this schedule can be run for very short time periods, the results being fed back into the program as starting conditions for the next short time period. The approach described here is less precise but more practical; material that shows a cum cycle time from its current operation to operation n (including the cycle time of n) of less than the time period being planned is presumed to be available to operation n during the time period being planned.]

The objective is to generate a schedule that sets two activity numbers for operation n, one number that is presumed achievable, published as the plan for the week, and a second number that is what would have been needed to bring the line fully into balance, if there had been sufficient material predicted to be available to that operation.

The formulas for calculating required activity are now adjusted as follows, where AP is activity planned, and AR is activity needed to achieve line balance:

$$AR_n = (AR_{n+1} + TWIP_{n+1} - WIP_{n+1}) / Y_n, \text{ and}$$

$AP_n = AR_n$, but not to exceed available material during the time period being planned.

, Operation	1	2	3	4	5
$_n$, Predicted Yield	90%	95%	90%	98%	90%
$_n$, Required Activity (steady state)	295	265	252	227	222
$_{n+1}$, Required outs of n	265	252	227	222	200
T_n, cycle time (days)	5	2	3	5	1
T_n, cycle time (weeks)	1	.4	.6	1	.2
$WIP(EFG)_n$	200	80	120	200	40
CY_n	.68	.75	.79	.88	.90
WIP, target work-in-process	295	107	152	225	44
IP, actual work-in-process	200	50	100	300	5
R_n, Required Activity (line balancing)	372	278	213	266	222
aterial available		170	150	300	145
P_n, Planned Activity (feasible line balancing, 1-week schedule)	372	170	150	266	222

Figure 5

By this method the lack of material at any particular operation or set of operations within a process is not allowed to depress artifically the level of activity scheduled at preceding operations.

6. The production plans thus produced for each individual process can once again be summarized by operation and equipment type used, and this can be compared to available equipment and labor capacity. Should the proposed plan exceed either of these capacities or insufficiently use either of these capacities, the required production output demand can be revised, or the proposed plan can be once again factored by the required capacity/available capacity.

The results of this exercise can be shown as required activity by operation. As long as there are not serious differences between the predicted yield and the actual observed yield for the activity taking place after the production plan is created, the plan remains the most useful pathway to balancing the line, and the goal of the dispatch list, in addition to other functions that it must serve, is to optimize performance to this plan. Where there are serious differences observed between actual and predicted yield the only practical approach is to change the predicted yield as needed to reflect a better estimate, and re-run the scheduling program. Self- correcting planning programs are very interesting but have no inherent advantages over the rerunning of the system described above, and they therefore imply a misuse of development and support resources.

CONSTRUCTING A SINGLE-OPERATION DISPATCH LIST

Now that, for any time period of interest, the exact production that is needed at each operation for each process being run has been determined, attention is turned to the functions of the dispatch list.

The dispatching function exists to determine the next material to which labor and equipment resources should be applied. Since the workplace is the scene of much coming and going of material, the dispatch list needs to be updated frequently. The actual choice of material must be made in consideration of at least the following factors:

* the priority of the lot of material,

* technical limitations on the allowable interval between the completion of one operation and the beginning of the next,

* grouping of lots that use the equipment in the same way so as to optimize the equipment capacity utilization,

* user preference of dispatching within each priority level (such as FIFO, LIFO and several others, out of which essentially only one option can be chosen at any time), and, of course,

* line balance.

Let us first examine the dispatching function without respect to line balance, as the line balancing function will depend on this as an underlying structure. By definition an operation may be used more than once in a given process, and several operations may use the same type of equipment. Obviously the proposed system must be explicitly aware of the equipment being used for each operation.

The first problem to be solved is the determination of what lots comprise the set of lots available for ordering into a dispatch list. In an environment of finite computing resources, it is not sufficient to simply include the lots awaiting work at a given operation, because it will not be practical to rerun all of the programs necessary to order correctly the dispatch list every time a lot moves from one operation to another. And, since there is not likely to be any great interest in elaborate and sophisticated dispatching schemes where a FIFO approach is rigidly enforced, one cannot argue that lots which arrive during the working shift simply be put at the bottom of the dispatch list. One may include on the dispatch list, for operator convenience, any lots that are currently being processed at this operation, but this is a matter of personal style or preference. Obviously, all material currently awaiting processing at this workstation must be included in the set of lots to be ordered on the dispatch list. Further, the list should include all lots that are likely to arrive at this workstation before the next dispatch list is generated. For the purposes of discussion, presume this dispatching interval to be one working shift, about 8 hours; it could be any time period from, say, fifteen minutes to a full working week, depending upon the computing resources available, the typical cycle time of an operation, and the amount of unpredictability in the environment.

Lots that are likely to arrive can reasonably be defined as lots that are now queued or active at operations prior to the operation to be dispatched, where the cumulative predicted cycle time between the current location and the operation to be dispatched is less than the interval on which new dispatch lists are run. Since the negative effects are greater when lots arrive that do not show on the dispatch list than when lots on the list do not arrive during the interval being dispatched it is advisable to err on the side of optimism in terms of assumptions about cycle times. Reasonable assumptions could include using the predicted cycle time of the highest priority recognized by the system (ignoring the actual priority level associated with the lot), presuming that the lot will leave the current operation immediately (or, if predicted cycle times are broken down into predicted wait times and standard processing times, presuming that lots now being processed will complete at exactly the start time + standard processing time and that all remaining lots will then start processing).

Having established the set of lots to be ordered at a given operation, and presuming that the priority of the lots is under good control of the master planning system or competent human intervention, the actual ordering of the lots to be worked on - AT THIS OPERATION - will have relatively little effect on the problems posed by the repeated use of a single workcenter in the manufacturing processes. What WILL affect line balance is the choosing, at a given piece of equipment, between the various specific process steps or operations that may be performed by this equipment. Much work has apparently been done at this level, and mechanisms that appear, to this reader, to be powerful and to have been subjected to serious simulation and field testing have been reported. The following list could be amplified, but would still not address the issues that are central to this paper. Note that the

ordered list will include material not yet present (these need not be displayed to the operator, the automatic equipment, or transport systems until the lots actually arrive). The following dispatch order would be reasonable in most semiconductor wafer fabrication operations:

1. Lots currently running

2. Lots which would violate the stated maximum time between the previous operation and this operation if they did not start on the next cycle. If there is more than one lot on this list, the list should be ordered by the rules used below for all other lots.

3. In most facilities the remaining lots should be sorted by lot priority, but that is really a question of operating style and the purposes being served by the priority system. Within each priority, the lots could be ordered by any one of the following, which the user could preset, or choose at runtime:

   FIFO (used to enforce relatively uniform cycle times)

   LIFO (used when rapid feedback of technical information is needed to control previous operations - best choice is the most recent arrival)

   Number of days late or predicted late to lot-specific required out date (the required date out presumably having been set by some corporate planning system that has the capability to link specific external or internal orders for parts with specific lots of work-in-process)

   Ratio of amount of standard processing time left until the lot will be complete to the amount of time until the lot will be delinquent (Critical Ratio)

   Any other rules suitable to this operation

In all of the above cases, once a given lot is selected for inclusion on the dispatch list, any lots which use the same equipment in the same way (recipe) should be listed with it, not because of their priority, but so that large batches, sized for the actual machine load capacity, can be processed together. Clearly the dispatching system must know the machine load capacity as well as have a concept of recipe. Recipe, used in this context, refers to the specific ways in which a piece of equipment is used on a particular part type or within a particular process. If each process is specified independently of the other process, then the scheduling system will not be able correctly to discern when material from two different processes can be run together. Only if the recipes are used as building blocks for the processes can this limitation be overcome.

Lots that are currently on a hold status may be shown (as long as they are properly flagged as unavailable) or not, at the preference of the operating facility. They need to be included in the dispatching activity so their proper spot on the dispatch list can be known as soon as they are released from hold.

THE PROBLEM NOT YET SOLVED

The solution presented above is a reasonable structure for weighing the required delivery date of product, the maximum allowable time between process steps and, in a strictly linear flow process, inventory costs and equipment utilization. Unfortunately, it does not address a key feature of some manufacturing environments, semiconductor wafer fabs notably included. The problem, referred to earlier and here addressed more fully, involves the repeated use of the same equipment in the manufacturing process. A simplified view of this is shown in figure 6.

The key problem with using normal dispatching techniques in a production area that uses the same equipment or workcenter more than one time, is that it is unlikely that dispatching schemes that rely on the rules given for the creation of a single operation dispatch list will adequately feed all of the subsequent equipment. Note in the realistic but very simplified diagram below that the output of the masking line directly feeds four different equipment types.

If the output of this line does not feed each distinct type of equipment properly there will be a starvation of the equipment compared to the work that actually must be done at that station. Unless the quantity of work-in-process is increased to prevent this otherwise idle equipment time, the entire manufacturing facility will be underutilized. Assuming that the process is a continuous FIFO flow within the masking line, the critical dispatching point for this facility is the staging area at the beginning of the masking line. Failure to control this dispatching point can result in three out of the four subsequent workcenters being fed insufficient work; this will happen when, for example, a large group of 'early' lots arrives from one piece of equipment and those lots are, per FIFO, dispatched into the line without proper attention to line balance. Attempts to shorten overall cycle times by running just-in-time scheduling in such areas increases the likelihood of machines out of work and increases the need for a more sophisticated dispatching approach.

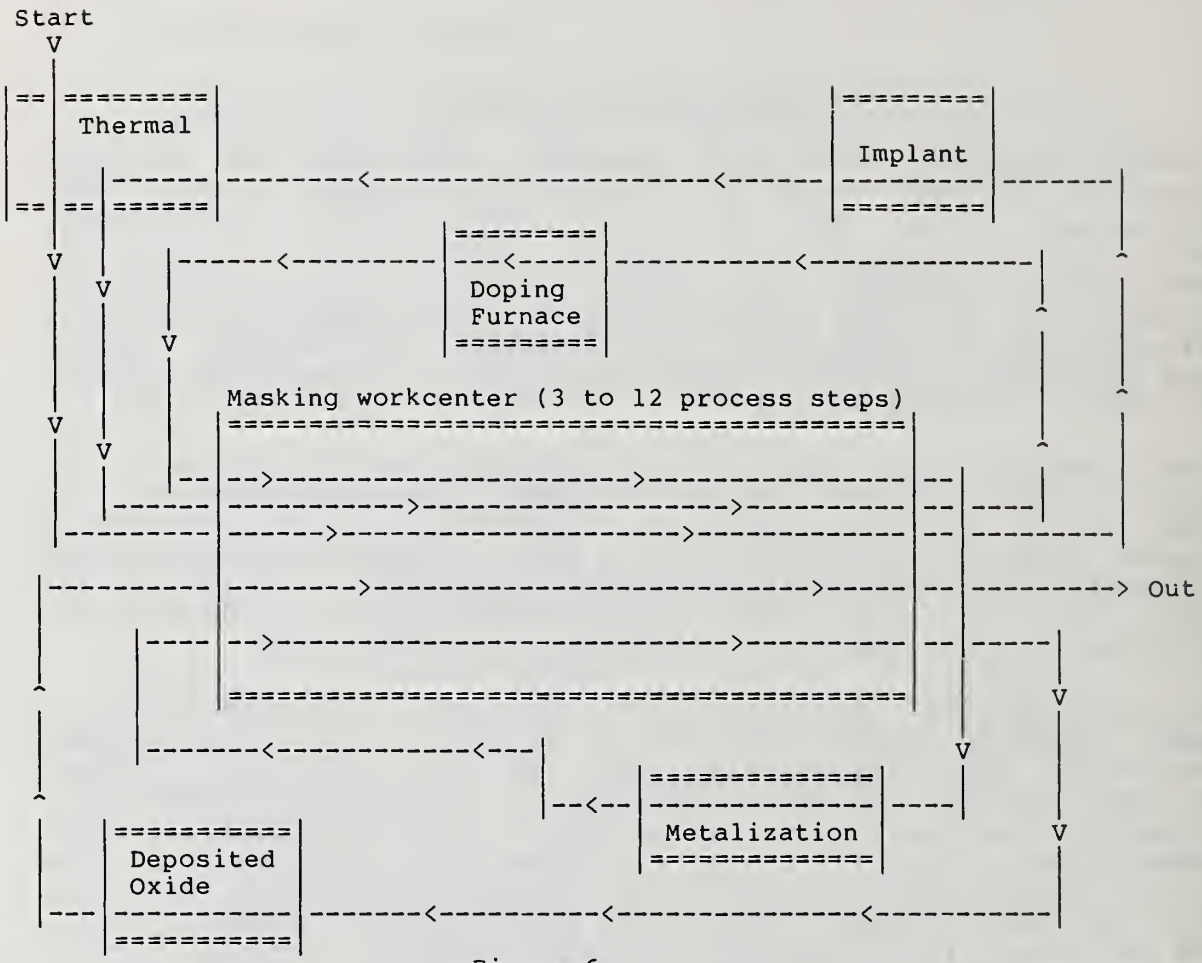


Figure 6

CONSTRUCTING A LINE-BALANCING COMPOSITE DISPATCH LIST

Having defined how each individual operation can be dispatched correctly, and having seen that the resultant dispatch list is insufficient for the particular line-balancing problem at hand, the critical issue is how to combine the dispatch lists of the various operations served by the same type of equipment. The operating principle is reasonably straightforward: the dispatch list should order the lots so that, if the dispatch list is followed exactly, all of the operations served by this equipment will end the dispatching period

(nominally one shift) equally ahead of or behind the planned work schedule. There is an underlying assumption here that, given a previously generated schedule of work that will bring the line into balance, and given that there have not been observed yields significantly different than predicted, the best pathway to balance the line is to perform the schedule.

This approach is not often found in commercial systems because the development efforts of the planning systems are often different than that of the actual tracking systems, making it very difficult to determine current performance to schedule from within the planning systems. Nevertheless, the line cannot reliably be driven into balance without this capability.

Dispatch lists can be combined where operations have common equipment; they can also be combined for an entire subdepartment within a production area for the purposes of assigning labor to the equipment that has material most urgently in need of being processed.

The combination of the dispatch lists from individual operations involves assessing for each operation the ratio of the work that has been actually completed to the work that is needed to be complete by the end of the current dispatching period. The first lot onto the composite list is the top lot on the individual dispatch list of the operation that is currently furthest behind schedule, since it is this lot that is most needed to balance the line. The ratio of this operation is then recalculated as if this first lot had already been processed with predicted yield. The next lot for the composite list is now chosen from the individual list of the operation that is furthest behind schedule; this may be the same operation as for the first lot on the list, or it may be another. This process continues until the dispatch lists of all of the affected operations are exhausted and the line approaches balance.

As a practical matter the top portion of the composite list can be built from only the lots currently being processed, exhausting all of these before moving on to deal with the lots in need of processing to avoid violating the maximum time between the completion of the previous operation and the beginning of the operation being dispatched. If lot priority plays an important role in the running of the production area, each level of priority from all of the dispatch lists of the individual operations can be exhausted before moving into the next lower priority.

The effect of this dispatching approach will be to leave all of the operations served by this equipment type or workcenter equally ahead of or behind schedule, which is the best that can be done for the line balance in the time allotted.

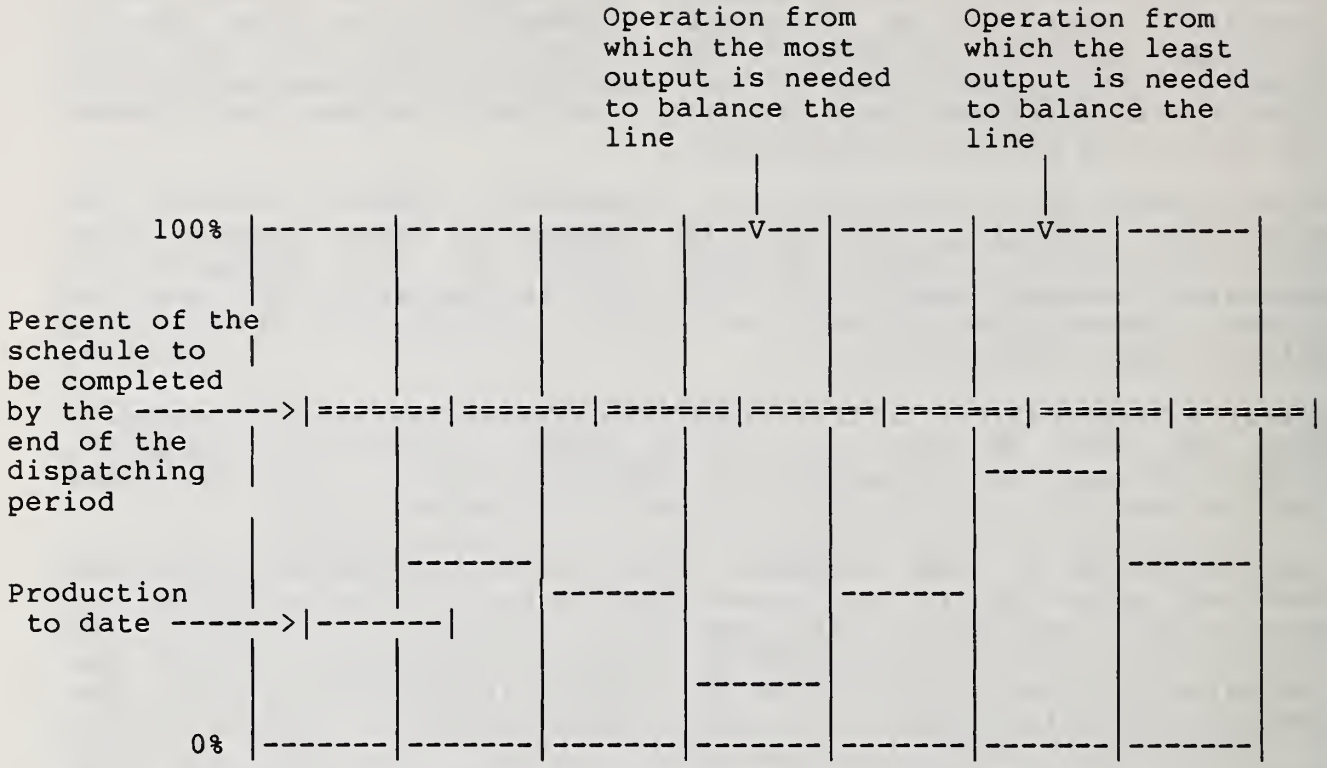


Figure 7

DISPATCHING AROUND EQUIPMENT THAT IS CURRENTLY DOWN

There is a great deal of concern in the semiconductor industry about the ability of dispatching systems properly to take into account the status of equipment that is downstream of the operation being dispatched, so that the lots that are moved through one operation can continue to move through the line, rather than accumulate in front of a piece of equipment that is not currently functioning. Comparison of the typical U.S. and European approaches to manufacturing with the very successful Japanese semiconductor manufacturing techniques suggests that the solutions to this problem lie in selecting and maintaining equipment and running that equipment within process limits that it will tolerate and support over long periods of time, so that the incidence of unpredicted unavailability is extremely low. To the extent that this concern for dispatching material around equipment currently down arises from a desire to minimize cycle time, it is worth noting that the most successful stories in the industry usually involve stopping production until all critical equipment is ready to function. While this admittedly does not involve a high degree of operations research finesse, it does lay the problem squarely at the feet of those responsible for selecting and maintaining the equipment, and, if the learning curve can be endured, seems to produce more predictable equipment availability. Failing proof of more appropriate solutions to this problem, the following is offered:

Excluded from consideration here are manufacturing environments in which it is a routine matter to have several pieces of critical equipment unavailable for production at the same time. In such an environment the only apparent solution is to run the equipment that is up, following the guidance of the dispatch list as closely as possible.

In relatively more controlled environments, where, at any point in time, only a very few pieces of equipment may be down, there do seem to be some techniques that are of some help. If, for instance, the down equipment is not used in all of the processes being run, then those processes that are not affected should be run at their maximum capacity, and the material that is running on the processes that are affected should be heavily deprioritized or shut down altogether. Unfortunately, the differences between processes in at least the semiconductor industry more often involve the number of times that a piece of equipment is used, or the way in which it is used, rather than the use or non-use of a specific piece of critical equipment. Furthermore, this is especially true of the larger volume production facilities. Cycle time can be very important to the smaller and research and development facilities, and this technique can be important in some applications, but it is no panacea.

Where the offending equipment is used in essentially all of the processes being run, the following should be true. If the equipment, when it is up, has more capacity than is needed for the current schedule, the line should continue to run, piling the material in front of the down equipment. Provision should be made to work overtime at selected operations once the equipment is once more functional, or to borrow equivalent capacity from some reasonable source. Where the equipment does not have capacity beyond that required for the current production schedule, shut the line down and return to work once the critical equipment is again functional.

SUMMARY

The precision with which dispatching must be done increases with the level of automation. The dispatching function must be tied to a realistic interpretation of real demand on the corporation. In most environments there is insufficient computer power to dynamically recalculate the dispatch list each time a lot moves, so some type of forward looking dispatching is needed. The ability properly to dispatch equipment or workcenters that are used more than once in the processes being run depends, as a practical matter, on having an exact knowledge of current actual production compared to the amount of production that is needed by this time. Until there is reason to re-calculate the dispatch list, the order of the composite dispatch list is a direct function of the extent to which each specific operation is lagging its to-date level of required activity.

VITAE: Henry Watts is Vice President of Semiconductor Operations, PROMIS Systems Corporation. Before joining PROMIS he spent 15 years in the semiconductor industry, primarily in manufacturing operations. He studied at UCLA, earning a BA in Psychology in 1967 and an MS in Socio-Technical Systems from the Graduate School of Business Administration in 1968. He lives in Sunnyvale, California.

SCHEDULING JOBS IN FLEXIBLE MANUFACTURING SYSTEMS

Ali S. Kiran
University of Southern California
Department of Industrial and Systems Engineering
Los Angeles, CA 90089-1452

Sema Alptekin
University of Missouri - Rolla
Department of Engineering Management
Rolla, MO 65401

1. INTRODUCTION

Flexible Manufacturing Systems (FMS) are automated batch manufacturing systems that require a high level of coordination and control of manufacturing activities. Flexible Manufacturing Systems are designed to produce a variety of part types simultaneously, in small lot sizes. The components of an FMS are NC machine tools, robots, automated inspection and in-process storage areas, and an automated material handling system all under control of a hierarchical computer system.

Flexible Manufacturing Systems are developed to take advantage of flexible automation. In contrast to fixed automation, FMS's are characterized by almost nonexistent changeover times from one part type to another. Hence, the batch sizes can be reduced without losing the economical advantages of fixed automation. Smaller batch sizes and simultaneous production of different part types, in turn, will result in shorter production lead times, improving the due date performance and customer/user satisfaction. In addition, other benefits of fixed automation such as part interchangeability and quality are still valid in flexible automation.

Design, planning, scheduling and control issues, on the other hand, become more difficult in flexible manufacturing systems than in conventionally equipped or fixed automated manufacturing systems. Compatibility of machines, fixtures and pallets, coordination of robots, machines and material handling equipment are some of the major concerns for FMS users and FMS manufacturers. Another concern for FMS users is lack of proper software to maximize efficiency of FMS's. Currently available operating software for FMS is unable to use sophisticated operations managements techniques. In fact, lack of proper coordination between the resource and time allocation decisions is among the major reasons for FMS down time and inefficient utilization of expensive hardware.

In this paper, we shall focus our attention on the FMS scheduling problem. We first give a brief background and definitions of decision issues in the life cycle of an FMS. We then present a heuristic scheduling algorithm to minimize tardiness in FMS's. Possible extensions of the scheduling algorithm and directions for future research will be discussed in the last section of the paper.

2. BACKGROUND AND LITERATURE REVIEW

The following issues have to be addressed at different stages of the FMS's life cycle: design, aggregate planning, system setup, scheduling, and control.

Design problems include initial specification and system configuration decisions such as availability and arrangement of machines, material handling equipment, storage spaces, computer(s), and control units. If the FMS under consideration is a part of the larger manufacturing environment, integration of the FMS with factory-wide production should be taken into consideration at the design stage.

Aggregate planning refers to medium range decision problems such as determination of production and material requirements, development of a master production schedule and coordination of activities in the entire factory. The built-in responsiveness of an FMS to variations in external demand makes it possible to delegate some classical aggregate planning decisons to the system setup stage.

System setup refers to a segment of the master schedule for which certain resource allocation decisons have to be made [7]. These decisons are:

1. Part Type Selection: Determining a set of parts to produce in the system setup period.
2. Tooling: Assigning required tools to machines.
3. Fixture Allocation: Allocation of the limited number of fixtures to part types.
4. Operations Assignment: Assigning operations to machines which have been equipped with the proper tools.
5. Routing: Determining part routings in the system.

The solution of the system setup problems will yield
- the set of parts which will be produced during the setup period,
- a machine routing for each such part
- an allocation of tools to machines which will achieve the production goals set by the master schedule
- an allocation of fixtures to parts, and
- an assignment of operations to machines.

After solving the setup problems, the next task is to determine start and completion times for each activity. We shall refer to this stage as the scheduling stage. The scheduling problem will be defined later in this section. The solution of the scheduling problem is input to the FMS controller, which is responsible for the next stage.

The control stage deals with the actual operation of the system. The problems at this stage include determination and implementation of policies to handle machine tool and other breakdowns, periodic and preventive maintenance, and quality and quantity control of in- process and/or finished goods [14].

A more precise definition of the FMS scheduling problem may be given as follows: Given an FMS, its actual state, and a set of parts

with known processing requirements and due dates, determine the start and completion time of operations of each part to be produced. The FMS scheduling problem can be approached from two different views:

1. Scheduling all operations of available jobs at the beginning of a predetermined scheduling period, i.e.,a priori scheduling.
2. Schedule operations one at a time when they become available, i.e., on-line dispatching.

The problem definition given above is valid for both a priori and on-line scheduling problems. On-line scheduling, however, is considered as a part of the control stage in most of today's FMS's. This is due to the difficulties encountered in implementing on-line scheduling algorithms with the look ahead feature.

The FMS scheduling problem is a more general case of the well known job shop scheduling problem [8]. It has been shown that FMS scheduling problems are at least as hard as job-shop scheduling problems. This is caused by additional resource constraints in the FMS scheduling problem, such as fixture and pallet availability and limits on automated material handling system and in-process storage space availabilities. In addition, flexible part routings and alternative machines further increase the number of alternative feasible schedulules and computational requirements of solution algorithms. The mixed integer program developed by Chang and Sullivan [1] is a good example to demonstrate complexity of the problem. For a 5 workstation - 10 part type FMS scheduling problem, their model has 30,000 binary decision variables, 500 continiuos decision variables, and 25,000 constraints.

The literature on FMS scheduling problems is sparse. Morton [10] attempted to classify the scheduling issues on FMS's and described a four level hierarchical decision structure in which both a priori and on-line scheduling have been recognized. The need for seperating system setup and scheduling issues is also recognized in [7].

Erschler et al. [4,5] analyzed periodic release of parts into the system. Chang and Sullivan [1] and Chang et al. [2] developed a mixed-integer programming model and an approximate solution method for operations assignment, routing and scheduling problems. Tang [15] developed a job scheduling model for a single machine FMS problem to minimize the number of tool replacements.

Dispatching rules have been developed and tested by Hutchinson and Wynne [6], Stecke [13], Lin and Lu [9], and Wang [16]. Generalization of these experimental results, however, is even more difficult in FMS's than in conventional job shops.

A heuristic scheduling method for FMS scheduling problems is presented in the next section. The proposed algorithm is based on sampling technique, and is applicable to most of the scheduling criteria. The algotithm is, however, implemented to minimize the total tardiness. This is due to the importance of such criteria in industrial scheduling problems. Panwalkar et al. [12] reported that meeting due dates is the most important goal, followed by minimization of setup

times and minimization of in-process inventory (in that order) in industrial scheduling. Yet relatively little has been done for tardiness in realistic environments due to its inherent complexity [11].

3. A HEURISTIC ALGORITHM FOR SCHEDULING JOBS IN FMS

We have developed an iterative heuristic method which generates active schedules using the priority function

$$z_i^k = \alpha\ f_i^k + (1-\alpha)\ f_i^{k-1} \qquad (1)$$

where

z_i^k: priority of job i at iteration k,
f_i^k: scheduling criterion value of job i at iterarion k,
α : a constant.

The algorithm starts with an initial active schedule which may be obtained using any dispatching rule. It then updates part priorities, based on effectiveness of the current schedule, using (1) and reschedules the jobs if updated priorities are different then priorities at the previous iteration and an improvement is possible over the current schedule. The algorithm stops if the performance of a schedule cannot be improved in n iterations where n is a user defined parameter. The development of the heuristic to minimize total tardiness is given below.

Let I^o be the set of parts (jobs) which are to be scheduled at time t=0. Let d_i be the due date and C_i be the completion time of job i. Let J_i^o be the set of operations of job i to be scheduled on the FMS. Tardiness of job i is defined as

$$T_i = \max \{ 0 , C_i - d_i \} \qquad i \in I^o \qquad (2)$$

and total tardiness of any schedule can be defined as

$$\Sigma T = \sum_{i \in I^o} T_i \qquad (3)$$

Job tardiness values are also used for updating job priorities

$$z_i^k = \alpha\ T_i^k + (1-\alpha)\ z_i^{k-1} \qquad (4)$$

where z_i^k is the updated priority of job i, and z_i^{k-1} is the current priority of job i.

The algorithm schedules the parts in non-increasing order of z_i^k values. Hence job i^* will be selected from the set of available jobs, $I \subset I^o$, and will be scheduled next if

$$z_{i^*}^k = \max_{i \in I} \{ z_i^k \}. \qquad (5)$$

The pseudocode for the algorithm is given as follows

procedure TARDINESS
  STOP = 0; TOLD = ∞; COUNT = 0; I = I^o

```
  do A while STOP = 0
    do B while I ≠ ϕ
      Update prioritities for all jobs, i∈I
      Select highest priority job i*
      J_i* = J_i^o*
      do C while J_i* ≠ ϕ
        Select the first unscheduled operation, j ∈ J_i*
        Calculate start and completion times
        Update resource availabilites
        J_i* ← J_i* - { j }
      enddo C
    UPDATE JOB TARDINESS
    I ← I - { i* }
  enddo B
Calculate total tardiness for the new schedule, TNEW
if TNEW < TOLD then record the new schedule
                else COUNT = COUNT ← 1
endif
if COUNT > n or TNEW = 0 then STOP = 1
                         else I ← I^O
endif
enddo A
end TARDINESS
```

4. IMPLEMENTATION OF THE ALGORITHM

The tardiness algorithm has been implemented in a scheduling system. The scheduling sytem requires the solution of the setup problem as input and provides information for on-line control of the flexible manufacturing system.

The overall structure of the scheduling system is given in Figure 1. The main program is a menu driven communication and control program. The desired functional program will be called by the main program after a selection has been made. The user has the following options:

1. Run the scheduling algorithm
2. Run the database management program
3. Perform a given schedule without interruptions

The database management program retrieves data required to schedule the part flow in the system. This can be done by transferring the data from a specified data file or entering the data directly. Reports on the status of the shop can also be generated using the database management program.

The scheduling program provides the following options to the user:

1. Schedule all available parts
2. Schedule only newly arrived parts
3. Reschedule parts in the case of machine failures.

In all three cases the scheduling algorithm will be performed in an appropriate way by adjusting inital job set I^o and resource availabilites. If a machine breakdown occurs at time t, the third

option of the scheduling system reschedules the remaining operations that require the failed machine during the breakdown period of Δt time units.

The third option runs the automatic shop control program. This program should be designed according to specifications of the particular FMS controller, which is responsible for sending proper commands to machines, robots and the material handling system, and for monitoring the system.

5. DISCUSSION AND FUTURE RESEARCH

The interaction between the system setup and scheduling stages needs further discussion. The solution of the setup problems specifies the resource allocation decisions using aggregate system performance measures. The solution also defines the feasible solution set of the scheduling problem. The solution of the scheduling problem, i.e. determination of operation start and completion times, provides a detailed view of the system performance which may not be acceptable by managers of the system, hence parameters of the system setup model may need to be readjusted until satisfactory solutions to both setup and scheduling problems have been obtained. The length of this sequential iterative process can be shortened by relaxation of the proper constraints in the system setup model. The resulting solution of the system setup model, in this case, will have greater scheduling flexibility. The scheduling problem, with the increasing number of alternative solutions, however, will also become more difficult. The determination of the balance between flexibility and responsiveness of the scheduling system is subject to further research.

We are currently investigating better feedback mechanisms for updating part priorities. This research may be extended to development of an adaptive feedback coefficient by defining $\alpha(t)$ as a function of scheduling performance at time t. The look ahead feature of the algorithm may be enhanced by application of AI techniques. For example, the scheduling flexibility may be adjusted in the case of heavy loads or tight due dates.

REFERENCES

1. Chang, Y. L. and Sullivan, R. S., "Real-time Scheduling of FMS", presented at TIMS/ORSA San Francisco Meeting, May 1984.

2. Chang, Y. L., Sullivan, R. S. and Bagchi, U., "Experimental Investigation of Quasi-realtime Scheduling in FMS", Proceedings of First ORSA/TIMS Conference on FMS, August 1984.

3. Denzler, D. R., Boe, W. J. and Duplaga, E., "An Experimental Investigation of FMS Scheduling Rules under Uncertainty", presented at TIMS/ORSA Boston Meeting, April 1985.

4. Erschler, J., Roubellat, F. and Thuriot, C., "Periodic Release Strategies for FMS", presented at TIMS/ORSA Meeting, San Francisco, May 1984.

5. Erschler, J. Roubellat, F. and Thuriot, C., "Steady State of a Flexible Manufacturing System with Periodic Releasing and Flow Time Constraints", Proceedings of First ORSA/TIMS Conference on FMS, 327-345, August 1984.

6. Hutchinson, G. K. and Wynne, B. E., "A Flexible Manufacturing System", Industrial Engineering, 10-17, December 1973.

7. Kiran, A. S. and Tansel, B. C., "Mathematical Models for Flexible Manufacturing Systems", Working Paper 86-01, Department of Industrial and Systems Engineering, University of Southern California, 1986.

8. Kiran, A. S., "Complexity of FMS Loading and Scheduling Problems", Working Paper 86-02, Department of Industrial and Systems Engineering, University of Southern California, 1986.

9. Lin, L. S. and Lu, C. Y., "The Scheduling Problem in Random Flexible Manufacturing Systems", Proceedings of First ORSA/TIMS Conference on FMS, August 1984.

10. Morton, T. E., "Scheduling a Flexible Manufacturing Cell", Carnegie-Mellon University, Pittsburgh, Pennsylvania, 1984.

11. Morton, T. E., Rachamadgu, R. M. and Vepsalainen, A., "Accurate Myopic Heuristics for Tardiness Scheduling", Working Paper 36-83-84, Carnegie-Mellon University, 1984.

12. Panwalkar, S. S., Dudek, R. A. and Smith, M. L., "Sequencing Research and the Industrial Scheduling Problem", Symposium on the Theory of Scheduling and Its Applications, 29-38, 1973.

13. Stecke, K. E. and Solberg, J. J., "Loading and Control Policies for a Flexible Manufacturing System", International Journal of Production Research, 19, 5, 1981.

14. Stecke, K. E., "Design, Planning, Scheduling and Control Problems of FMS", Proceedings of First ORSA/TIMS Conference on FMS, 1-7, August 1984.

15. Tang, C. S., "A Job Scheduling Model for a Flexible Manufacturing Machine", Working Paper 2/85, Yale University, 1985.

16. Wang, H., "An Experimental Analysis of Flexible Manufacturing Systems", presented at ORSA/TIMS Dallas Meeting, November 1984.

DESIGN REQUIREMENTS FOR A REAL-TIME PRODUCTION SCHEDULING DECISION AID

Frederick A. Rodammer, Graduate Research Assistant
K. Preston White, Jr., Associate Professor
University of Virginia
Department of Systems Engineering
Charlottesville, Virginia 22901

1.0 Introduction

Production scheduling is the allocation of available production resources over time to satisfy some set of production performance criteria (Graves; 1981). Typically, the scheduling problem involves a set of jobs to be completed, where each job comprises a set of operations to be performed. Operations require machines and material resources and must be performed according to some feasible technological sequence. Schedules are influenced by such diverse factors as job priorities, due-date requirements, release dates, cost restrictions, production levels, lot-size restrictions, machine availabilities and capabilities, operation precedences, resource requirements, and resource availabilities. Performance criteria typically involve tradeoffs between holding inventory for the task, frequent production changeovers, and satisfaction of production-level requirements and due dates.

Scheduling involves: (1) selecting a sequence of operations or process routing, the execution of which results in the completion of an order, and (2) assigning times (i.e., start and end times) and resources to each operation. Process routing selection is typically the product of a planning process, while the assignment of times and resources is typically the purpose of scheduling (Fox, et al.; 1983).

The efficient and effective scheduling of production lines represents a major obstacle to the success of current efforts in automated manufacturing. Despite extensive investigation, traditional methods of operations research have failed to yield a practical approach to the problem of production scheduling in large and complex manufacturing environments. Various modeling paradigms, i.e., combinatorial optimization, stochastic optimization, discrete-event simulation, commercial approaches, artificial intelligence, and control theory, have not yet provided an acceptable approach to the problem of rapid rescheduling of non-empty production lines in the face of a range of conflicting constraints and objectives.

In this paper, we propose design requirements for a production-scheduling decision aid which relies upon a synthesis of deterministic scheduling heuristics and discrete-event simulation. The purpose of this decision aid is to recommend schedules which improve production efficiencies and which accomodate the periodic need to reschedule non-empty production lines in the face of changes in demand and productive capacities. We begin our discussion of the scheduling problem by presenting an overview of the operational context of the scheduling problem and by describing the characteristics of the real-time scheduling environment. A summary and critique of various scheduling model paradigms which are used to develop solution strategies for the scheduling problem is presented in Section 3.0. Section 4.0 describes the proposed formulation of the scheduling problem, which incorporates the characteristics of the real-time scheduling environment. The simulation-

optimization solution strategy for the problem is discussed in Section 5.0. Development and implementation issues and directions for future research are identified in Section 6.0.

2.0 Operational Context for the Real-Time Decision Aid

Production scheduling is typically viewed as an intermediate function within a hierarchical framework of manufacturing decision making. In order to establish the context in which scheduling is undertaken, one such framework is discussed here. Using this framework, the essential elements of the real-time production scheduling environment are defined.

2.1 A Hierarchical Framework for Manufacturing Decision Making

We propose a five-level framework of manufacturing decision making to provide the context for the scheduling function. The hierarchical representation is similar to frameworks used in hierarchical control theory problems (Abraham, et al.; 1985, Akella, et al.; 1984, Gershwin, et al.; 1984, Gershwin, et al.; 1985, Hax and Meal; 1975, and McPherson and White; 1986). The hierarchical framework provides representation of the system planning, production planning, flow planning, scheduling, and processing functions.

System planning, the highest level of the decision hierarchy, specifies and organizes the manufacturing resources necessary to meet long-term production goals. The system planning function provides a facilities design which is used in production planning. The production planning function establishes aggregate production rates or volumes, consistent with the capability of the manufacturing system, in order to satisfy aggregate demand in an economic manner. Production planning supplies information to the next lowest level of the hierarchy, the flow planning function. Flow planning determines the sizes of production batches and the sequences in which these batches will flow through their process steps. Flow plans must be consistent both with the production plan (i.e., the aggregate production rate) and with resource constraints and demand requirements. Flow planning also includes the determination and location of protection stocks (e.g., WIP) necessary to buffer the flow plan from disruptions.

The scheduling function resides at the fourth level of the five-tiered manufacturing decision hierarchy. Figure 1 shows the inter-relationships between the flow planning function and the scheduling function, on the one hand, and the scheduling function and the processing function, i.e., the fifth level of the decision making hierarchy, on the other.

The upper portion of Figure 1 depicts information from the flow planning function, provided in the form of present and future flow plans which contain information on sizes of production batches and sequencing of process steps, being processed in the scheduling decision aid data structure. This flow planning information is combined with information from the processing function, e.g., shop-floor status information, which provides measures of the current in-process inventory levels and machine status of the schedule currently in operation.

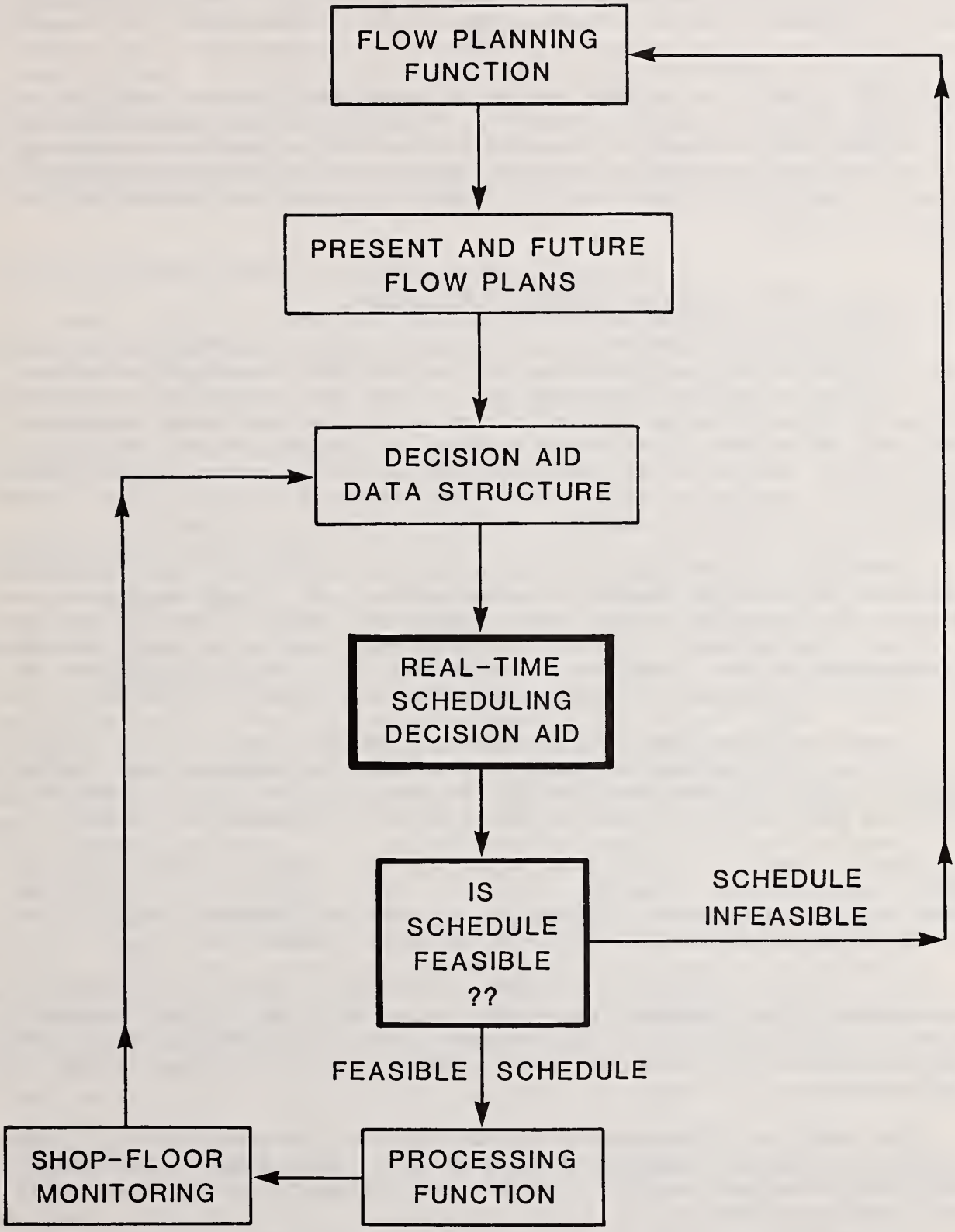


Figure 1: OPERATIONAL CONTEXT FOR THE REAL-TIME SCHEDULING DECISION AID

Events such as "hot" jobs, machine failures, or slippage in due-dates require the generation of a new schedule. This scheduling (re-scheduling) process is done in the environment of non-empty production lines. The lower portion of Figure 1 depicts a new schedule, which addresses the most current flow planning and shop-floor status information, being generated in real-time. The schedule is checked for feasibility and if the schedule is feasible and acceptable, then it is implemented and the resulting processing function is performed according to the new schedule. If the schedule is infeasible, then the current flow plan must be augmented in order that a feasible schedule may be generated. The real-time scheduling process continues in a dynamic fashion as new schedules are "fit" into the current operating environment of the production facility.

2.2 The Real-Time Production Scheduling Environment

To a great or lesser degree, every scheduling environment is unique. We present a generalized conceptual view of the real-time scheduling environment here in order to provide a framework for our real-time production scheduling problem formulation presented in Section 4.0. The real-time production scheduling environment consists of a number of characteristics, which are listed below. Many of these characteristics are either ignored or simplified by classical "job-shop or machine-scheduling" problem formulations (i.e., the combinatorial optimization modeling paradigm described in Section 3.0).

The real-time production scheduling problem formulation must consider situations in which job inventory levels may be at any initial state, i.e., job inventory levels are not all zero, and initial states of the machines may be either busy or idle. Similarly, the problem formulation must address circumstances in which in-process inventory levels at the final state are not all zero and all jobs are not completed at some undefined final time.

In many real-time scheduling situations a job is broken up into a number of batches. The problem formulation, therefore, should be able to handle variable batch sizes and should be able to cope with a variable, interrupted processing time and batches that can be split by the schedule if necessary.

Setup times must be considered explicitly as a seperate entity from the fixed, uninterrupted processing time of a job. Set-ups in a real-time scheduling environment are a seperate entity from the processing time of a job and need to be considered explicitly in the problem formulation.

The job-shop scheduling problem formulation should be capable of representing job and resource unavailability over any time interval, including delayed release times, machine failures, operator unavailabilities, and material stockouts. Many real-time situations reflect these types of "disturbances".

The real-time scheduling environment includes the possibility of high priority jobs being suddenly introduced into the system. The problem formulation should, therefore, be capable of representing rush or hot jobs with higher priorities.

The formulation of the job-shop problem should be capable of representing flow plans in which a job is not necessarily processed on every machine and in

which the technological sequence may imply a semi-order on the machines and/or alternative process paths. In real-time scheduling situations, every job is not processed on every machine exactly once, in a strict technological sequence.

Finally, the problem formulation should be capable of representing multiple, conflicting criteria (minimize tardiness, minimize production and inventory costs, and maximize schedule stability) which can imply deliberate underutilizaion of machines to reduce inventories and/or insure reliability. Real-time scheduling objectives are numerous and are many times conflicting.

3.0 Scheduling Problem Research Areas and Model Paradigms

Six areas of research and practice which have been applied to various aspects of the scheduling problem are: (1) combinatorial optimization, (2) stochastic optimization, (3) discrete-event simulation, (4) commercial approaches, (5) artificial intelligence, and (6) control theory. Each of these paradigms offers insight into the fundamental scheduling problem, however, no single paradigm appears entirely adequate in addressing the operational context and real-time scheduling environment described in the previous section.

Combinatorial optimization, i.e., the classical job-shop or machine scheduling problem of operations research, has been studied extensively over the past several decades. The problem may be formally stated as follows (Bowman; 1959, Conway, et al.; 1967 and French; 1982): N jobs are to be processed on M machines. Each job consists of a set of M operations, one operation uniquely associated with each of the M machines. The processing time for an operation can not be split. Technological constraints demand that the operations within each job must be processed in a unique order. The scheduling problem involves determining the sequence and timing of each operation on each machine, such that some given performance criteria is maximized or minimized. Typical performance criteria include minimizing the makespan (i.e., minimizing the time required to complete all of the jobs) and minimizing maximum tardiness (i.e., minimizing the largest difference between completion times and due dates).

The job-shop scheduling problem is a highly simplified formalism for the production scheduling problem actually encountered in manufacturing. Nonetheless, the general job-shop scheduling problem is known to be NP-hard (French; 1984, Rinnooy Kan; 1976), i.e., the time required to compute an optimal solution to the problem increases exponentially with the size (number of jobs and number of machines) of the problem instance. Even modest problems (10 jobs and 10 machines) can not in general be solved to optimality, even with computing power that far exceeds the capacities of modern supercomputers.

Besides providing conclusive evidence that the production scheduling problem is difficult, work on the job-shop scheduling problem has generated a large number of scheduling heuristics. Heuristics are reasonable and computationally efficient rules for generating candidate schedules, which may or may not prove to be satisfactory in either the job-shop or production-scheduling environment. In general, heuristics select the next operation to be processed based upon some easily computed parameter of the jobs, operations, or machines. These parameters can include processing times, due

dates, operation counts, costs, setup times, arrival times, and machine loadings (French; 1982, Gere; 1966, and Mellor; 1966). Examples are SPT (shortest processing time first), LPT (longest processing time first), FIFO (first-in, first-out), LPR (longest processing time remaining), EDD (earliest due date), and pure random (Monte Carlo) selection. More complicated heuristics are generally built-up from simpler rules. Panwalkar and Iskander (1977), for example, cite some 113 scheduling heuristics that have been proposed or actually applied.

Work on the job-shop problem has also provided a wealth of information on solution strategies and approximation algorithms for determining acceptable schedules within a reasonable amount of time. In particular, partial enumeration techniques which combine heuristics and neighborhood search strategies have been shown to work reasonably well under various conditions. These strategies involve the use of a heuristic to find a good seed or starting schedule, modifying the seed, and evaluating the resulting schedule. A cycle of adjustment and evaluation is repeated until no further progress relative to the performance measure is achieved.

Although these contributions are significant, the job-shop scheduling problem itself is too restrictive a formulation to provide results that are anything more than suggestive for actual production scheduling. Developing an appropriate model for direct use in solving production scheduling problems will require a significant relaxation of the basic assumptions of the job-shop model, (i.e., refer to the characteristics of the real-time scheduling environment presented in Section 2.2).

Stochastic optimization addresses a number of types of manufacturing systems problems. These problems include queueing (French; 1982 and Stecke and Solberg; 1985), reliability (Ross; 1970 and Ross; 1983), lot-sizing (Afentakis; 1985, Karmarkar, et al.; 1985, Lasserre, et al.; 1985, and Steinberg and Napier; 1980), and inventory theory (Kivenko; 1981 and Winters; 1962). In queueing theory, jobs arrive in a random process and queue until a machine is free, whereupon a job is selected from the queue and assigned to that machine according to some predetermined priority rule. Selecting a priority rule which leads to the least expected cost of a job is, unfortunately, a very difficult task. Reliability theory is concerned with "maintaining" a stable schedule in lieu of machine failures occuring. Inventory theory and lot-sizing techniques attempt to determine lot sizes which will maximize production throughput and assure that due dates are met and also minimize production and inventory costs.

Discrete event simulation has been used primarily as a vehicle for testing fixed scheduling heuristics. A consensus among researchers appears to be that a combination of simple priority rules, or a combination of heuristics with a simple priority rule, works better than individual priority rules. This idea supports the concept of having a flexible simulation tool which can be used to schedule the job-shop. The user of the simulation tool reviews the status of the "job-shop model" which is dynamically displayed before him. He can use his knowledge and practical experience in order to stop, interact with the model and try alternative scheduling approaches (Hurrion; 1978, Shannon; 1984, and Trybula and Ingalls; 1985).

Scheduling simulation is designed to provide the user with the capability of performing "what-if" analysis on the current scheduling problem. The

scheduling options that can be explored experimentally are only limited by the time constraints and resources available to the user. Since the simulation process provides dynamic output, the user can identify long queues within the model and thereby attempt to eliminate the bottlenecks which may exist for priority jobs (Bulkin et al.; 1966).

An advantage of the simulation approach to scheduling is that it can model the effects of such factors as policy changes, which cannot be accounted for in an analytic model (Stecke and Solberg; 1980). Another advantage of discrete-event simulation is that it can provide the user with the opportunity of performing exploratory tests upon the schedules being produced (Baker and Dzielinski; 1960). Disadvantages of using simulation to produce schedules are that it is costly, both in the computer time used to generate schedules and in the human modeling effort required to design and run the simulation model.

Current commercial approaches can be characterized in terms of software products or philosophies. Manufacturing Resource Planning (MRP) (Fox; 1984), Just in Time production (JIT) (Schonberger; 1984), and Optimized Production Timetables (OPT) (Jacobs; 1984) are representative of the philosophies and associated computer software products that are currently in wide use.

Commercial software packages for MRP, OPT and JIT simultaneously address the production planning, flow planning, and scheduling functions. These are not distinct scheduling models and were not designed to specifically address the real-time scheduling problem. Developing a schedule for future operations given conditions on inputs, requirements for outputs, and resource constraints over time is a next to impossible task. It is analogous to attempting to solve the classic "two-point boundary value problem of control theory" (Abraham, et al.; 1985).

The two-point boundary value problem of control theory consists of an initial point, a final point, and trajectory constraints. The initial point is the problem inputs, i.e. the initial inventory levels and schedules of incoming parts and materials. The final point is the problem outputs, i.e. due dates for the products. The resource constraints (availability) are the problem's boundaries. All aspects of this problem are time-varying.

The two-point boundary value problem is computationally intractable, but it is possible to get a feel for the problem by relaxing one of the three conditions, i.e., the initial point, the final point or the boundary value. MRP systems choose to relax the boundary condition, OPT chooses to ignore the final point, and JIT chooses to ignore the initial conditions.

The artificial intelligence modeling paradigm depicts the scheduling problem as the determination and satisfaction of a large number and variety of constraints which are found in the scheduling domain (Fox, et al.; 1983). Artificial intelligence is used to extend knowledge representation techniques to include the variety of constraints found in the scheduling domain, to integrate the constraints into a search process, to relax constraints when a conflict occurs, and to diagnose poor solutions to the scheduling problem.

Artificial intelligence includes the areas of knowledge-based systems, expert systems, and learning systems (Herrod and Papas; 1985). Current research efforts in learning systems and genetic algorithms may hold potential for improving the solution speed and accuracy of the job-shop scheduling

problem. Learning systems and genetic algorithms may offer efficient search procedures for finding good solutions to computationally complex problems (Davis; 1985).

The control theoretic modeling paradigm seeks scheduling methods which either explicitly reflect the uncertain nature of the available information or give some guarantee as to the insensitivity of the schedule to future information. This modeling paradigm attempts to limit the effect of machine failures, operator absences, material unavailability, surges in demand, or other disruptions upon the scheduling process (Gershwin, et al.; 1984). Schedules which are robust to disruptions, robust to absence or inaccuracy of status information, and flexible to change are pursued by the control theorist (Abraham, et al.; 1985). The control theory paradigm views scheduling as a dynamic activity, and the scheduling problem is really one of understanding how to reschedule.

Although control theory has only recently been applied to discrete production scheduling, the underlying problem and fundamental issues are perhaps most naturally described as problems in control. Consider the standard control paradigm which is depicted in Figure 2 as the control system model for the scheduling problem.

The system under control operates on a sequence of inputs, u(t), yielding a sequence of outputs, y(t). The outputs at any time are a function of the state of the system, x(t). The system state at time t is defined as the minimum set of variables, such that knowledge of the state at some initial time, together with knowledge of the inputs to the system at this and all future times, is sufficient to determine (given an adequate model) all of the future states of the system. A general statement of the control problem is to determine a means to guide the system state (and thus the system output) through time according to some trajectory which satisfies the constraints imposed by the system model and which simultaneously satisfies some set of minimum performance criteria.

For a manufacturing facility, inputs to the system include:

Production orders, which specify the quantities of various jobs to be processed and the dates at which these quantities are due for delivery.

Raw material and labor, which are used in production and without which production is impaired.

Disturbance inputs, such as machine failures or labor outages, which alter the productive capacity of the plant over some period of time, but over which the scheduler has little influence.

Controlled inputs, such as scheduling, maintenance, and overtime decisions, which alter the productive capacity of the plant, but which the scheduler can regulate within certain bounds.

The state of a manufacturing facility at time t is the minimum set of variables the values of which completely define:

The levels of inventory for all completed and partially completed jobs at time t.

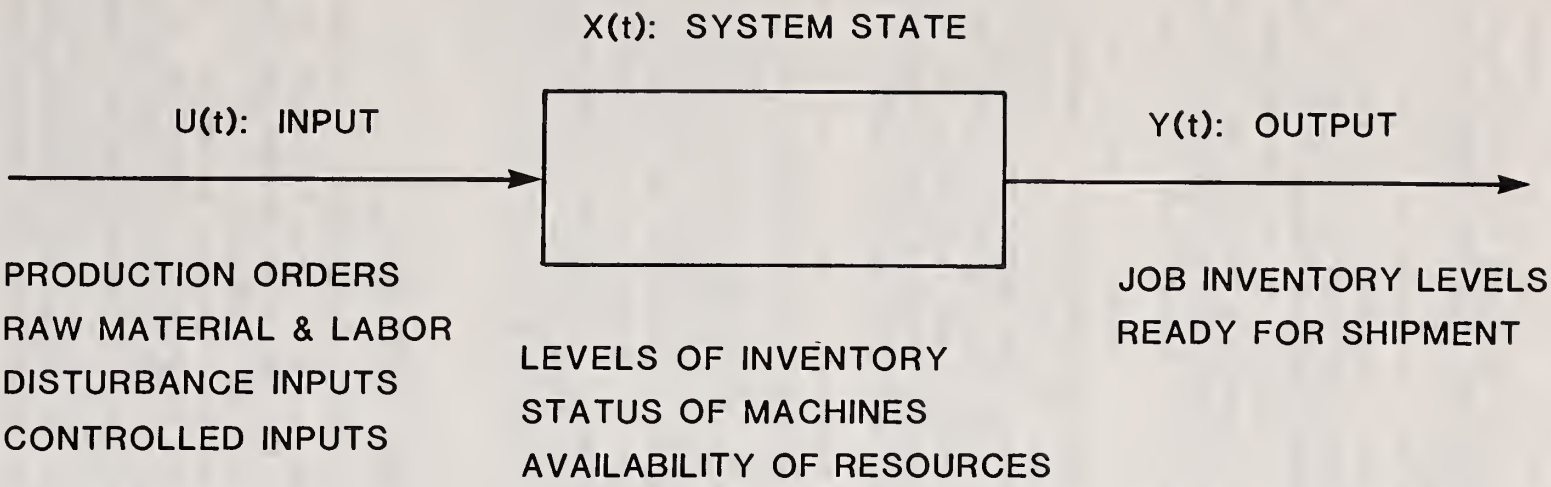


Figure 2: CONTROL SYSTEM MODEL FOR THE SCHEDULING PROBLEM

The status of all machines at time t (whether idle, active on job, in setup for job, or in repair).

The availability of labor and the levels of inventory for all materials at time t.

Outputs from the manufacturing facility can be any appropriate combination of the state, for example, the invetory levels of all jobs ready for shipment on a specified due date.

The scheduling problem is then to determine the control, i.e., the sequencing of the operations of each job, the scheduling of overtime and maintenance, and the timing of materials orders, which yields a desired state trajectory. The control must also satisfy constraints imposed by limited manufacturing resources and must simultaneously satisfy performance measures. These performance measures may include minimum tardiness of finished jobs, minimum and maximum in-process inventory levels, and minimum production and holding costs.

Merits of the control theory modeling paradigm are numerous. First, this modeling paradigm recognizes the need to integrate the scheduling activity with the planning activity. Second, the paradigm accepts the dynamic environment of the scheduling problem as a given and attempts to find schedules which are robust, flexible and adaptable to this dynamic environment. The control theory modeling paradigm also provides a wealth of knowledge in defining the real-time scheduling problem and the corresponding scheduling objectives.

While the control paradigm appears to be especially well suited to defining the fundamental problem qualitatively, control theory has yet to develop a set of techniques adequate to scheduling model formulation, analysis, or design. The mathematics and techniques of control theory apply to continuous systems (Markov processes) and are not well-adapted to discrete-event systems (semi-Markov processes). Those aspects of the manufacturing problem which can be usefully approximated by continuous systems are the areas most likely to benefit from traditional quantitative applications of control theory, at least in the short run. Production scheduling, unfortunately, does not appear to be one of these areas.

4.0 Formulation of the Job-Shop Scheduling Problem

In order to consider the most general situations and characteristics of the current-day scheduling environment, which were discussed in Section 2.2, we present a modeling approach for the real-time scheduling problem which incorporates a synthesis of the paradigms discussed in the previous section and which is based on the emergent control theoretic paradigm. We propose a multi-objective, mixed-integer, linear programming formulation of the job-shop scheduling problem (White and Rogers; 1985) which incorporates a finite time horizon, set-ups, overtime, buffer stocks, and shortfalls. The formulation considers initial and final states of the inventory levels as zero or non-zero and machine status as busy or idle. It allows for batch size and number of batches to be variable and it considers set-ups as explicit entities. Disturbances, job priorities, multiple machines or machine banks, and multiple optimality criteria are also considered.

4.1 Description of Variables and Parameters

Given the general requirements for the job-shop scheduling problem formulation which were listed in Section 2.2, we formulate the scheduling problem in its most general manner by including all possible constraints and objectives. Any implementation of this formulation may include only of subset of these. A description of variables and parameters follows:

Let t = 0, 1, 2, ..., T index a sequence of (equal) time intervals within a finite time horizon T. The problem's decision variables and parameter descriptions are defined as:

j — is the job number identifier; j = 1, 2, ..., J where J is the total number of jobs to be processed.

i — is the operation number; i = 1, 2, ..., I(j) where I(j) is the total number of operations associated with job j.

o(j,i) — is defined as the ith operation of job j.

p — is a unique machine operation; p = 1, 2, ..., P where P is the total number of unique machine operations.

m(p) — is the total number of machines which can process the a unique operation, operation p, of any given job j.

M(p) — is the set of all machines. Note that although a machine may be capable of processing more than one type of operation, it may not process more than one operation at one time.

x(j,i,t) — represents the number of machines processing o(j,i) during interval t, including machines being set-up to process o(j,i). x(j,i,t) takes on the following integer values:

$$x(j,i,t) = \begin{cases} 0, \text{ if } o(j,i) \text{ not in process during } t \\ 1, \text{ if 1 machine is processing } o(j,i) \text{ during } t \\ 2, \text{ if 2 machines are processing } o(j,i) \text{ during } t \\ . \\ . \\ m(p), \text{ if } m(p) \text{ machines are processing } o(j,i) \text{ during } t \end{cases}$$

where m(p), as defined above, is the maximum number of machines that are available to processs operation p during time interval t. In this definition, note that the ith operation of job j fits into the same category as the general operation p.

s(j,i,t) — is the number of machines "set-up" for o(j,i) during t.

Y(j,i,t) is the definition given to the in-process inventory for the ith operation of job j at time t.

p(j,i) is the process time for operation i of job j and is given in machines/unit.

f(j,i) is the fraction of a time interval required for a set-up of operation o(j,i).

V(p,t) is the overtime for M(p) at time t.

W(p,t) is the maximum overtime available for M(p) at time t.

d(j,i,t) is the amount of Y(j,i,t) desired at time t.

CS(j,i,t) is the cost per set-up s(j,i,t).

CV(p,t) is the cost per unit of overtime V(p,t).

CP(j,i,t) is the cost penalty for production shortfall, i.e., cost penalty when d(j,i,t) is greater than Y(j,i,t).

CH(j,i,t) is the holding cost for production surplus, i.e., WIP inventory cost when Y(j,i,t) is greater than d(j,i,t).

4.2 Hard Constraints

The restrictions for the job-shop scheduling problem are now defined. These restrictions are provided in terms of the machine number and overtime constraints, the machine set-up constraints, the in-process inventory definition, and the technological constraints.

The total number of machines which could process an operation is limited. This limitation is defined by the machine number (with overtime) constraints, condition (1). This condition applies to all jobs, a, processing the operations, b, where o(a,b) is defined here as the general operation, p.

$$o(a,b) \in M(p)$$

$$\sum x(a,b,t) \leq m(p) + V(p,t) \qquad (1)$$

where m(p) is the total number of machines which could process operation p out of the total set of machines M(p), o(a,b) is the specific operation p defined for all jobs, x(a,b,t) is the number of machines processing operation o(a,b) at time t, and V(p,t) is the overtime permitted for that set. The overtime permitted is limited by the maximum overtime available as denoted by the constraint given in condition (2). Conditions (1) and (2) apply to all p from 1 to P, all m(p) in the set of machines M(p), and all time periods, t, from 1 to T.

$$V(p,t) \leq W(p,t) \qquad (2)$$

We define the number of machine set-ups to be the number of machines which start to process o(j,i) in time interval t and were not processing o(j,i) in the previous time interval, (the first term in brackets in equation (3)), minus the number of machines which can process o(j,i) and were idle in the previous time period, (the second term in brackets in equation (3)). The set-up constraint is for all jobs, j, all operations, i, and for all time periods, t, from 1 to T.

$$s(j,i,t) = [\, x(j,i,t) - x(j,i,t-1) \,] - [\, m(p) - \sum^{M(p)} x(j,i,t-1) \,] \qquad (3)$$

The definition of in-process inventory, Y(j,i,t), at any time period t is the amount of inventory which has completed operation i of job j at time t. The expression for in-process inventory (equation (4)) is defined for all jobs, j, all operations, i, and for time periods, t, from 0 to T.

$$Y(j,i,t) = \sum_{k=0}^{t} [\, (x(j,i,k) - f(j,i)\, s(j,i,k)) / p(j,i) - (x(j,i+1,k) - f(j,i+1)\, s(j,i+1,k)) / p(j,i+1) \,] \qquad (4)$$

where p(j,i) is the process time per machine (machines/unit) and f(j,i) is the fraction of a time interval required for a set-up. Note that the initial states are specified by setting Y(j,i,0) and x(j,i,0) equal to their initial values and that the values for s(j,i,t) and x(j,i,t) at the undefined operation I(j)+1 are assumed to be zero.

Condition (5) enforces the technological constraints of the job processing order. These constraints are defined for all jobs, j, all operations, i, and for time periods, t, from 1 to T, except for the instances when i = t = 1.

$$Y(j,i-1,t-1) - x(j,i,t) / p(j,i) \geq 0 \qquad (5)$$

4.3 Objectives (or Soft Constraints)

We now provide representation for the objectives of the problem formulation. The first objective is minimize the lateness of an order and minimize total tardiness. We formulate this objective by considering the desired finished inventory for job j which is given as d(j,I(j),t). We wish

to have Y(j,I(j),t) as close to d(j,I(j),t) as possible in order to meet the finished inventory demand level at time t. Since this objective is formulated in relation to a desired inventory level, (i.e., finished inventory), we may incorporate it within the "minimize WIP inventory " objective which we will now define.

To minimize the WIP inventory, we desire to have Y(j,i,t) as close as possible to the desired inventory level, d(j,i,t), for all j, i, and t. If Y(j,i,t) is less than d(j,i,t), then we have a shortfall or do not have enough buffer stock. If Y(j,i,t) is greater than d(j,i,t), then we have a production surplus, (too much buffer stock) and have increased WIP inventory. Thus, to minimize inventory, we wish Y(j,i,t) to follow the "desired trajectory" defined by d(j,i,t). The objective to minimize shortfalls and minimize surplus is specified in (6) as:

$$\text{Minimize:} \quad \sum_{k=1}^{T} \sum_{j=1}^{J} \sum_{i=1}^{I(j)} \left| Y(j,i,k) - d(j,i,k) \right| \qquad (6)$$

To minimize set-up and change-over costs, we minimize the number of set-ups multiplied by the cost per set-up. The objective to minimize set-up costs is specified in (7) as:

$$\text{Minimize:} \quad \sum_{k=1}^{T} \sum_{j=1}^{J} \sum_{i=1}^{I(j)} CS(j,i,k)\ s(j,i,k) \qquad (7)$$

To minimize with respect to overtime, we minimize the amount of overtime multiplied by the cost of overtime. The objective to minimize overtime costs is specified in (8) as:

$$\text{Minimize:} \quad \sum_{k=1}^{T} \sum_{p=1}^{P} CV(p,k)\ V(p,k) \qquad (8)$$

To minimize with respect to costs of shortfalls and holding inventory, we minimize with respect to the amount of shortfall multiplied by the cost penalty for the shortfall and add to this the amount of holding inventory multiplied by the holding cost for the surplus inventory. Our objective to minimize shortfall and surplus costs (minimize inventory costs) is specified in (9) as:

(9)

$$\text{Minimize:} \quad \sum_{k=0}^{T} \sum_{j=1}^{J} \sum_{i=1}^{I(j)} [\, CP(j,i,k) \max \{ 0, (d(j,i,k) - Y(j,i,k)) \} + CH(j,i,k) \max \{ 0, (Y(j,i,k) - d(j,i,k)) \,]$$

If we wish to minimize total costs, then the corresponding objective is specified in (10) as:

(10)

$$\text{Minimize:} \quad \sum_{k=1}^{T} \sum_{p=1}^{P} CV(p,k) \; V(p,k)$$

$$+ \sum_{k=1}^{T} \sum_{j=1}^{J} \sum_{i=1}^{I(j)} CS(j,i,k) \; s(j,i,k)$$

$$+ \sum_{k=0}^{T} \sum_{j=1}^{J} \sum_{i=1}^{I(j)} [\, CP(j,i,k) \max \{ 0, (d(j,i,k) - Y(j,i,k)) \} + CH(j,i,k) \max \{ 0, (Y(j,i,k) - d(j,i,k)) \,]$$

The multi-objective, mixed-integer, linear programming formulation of the job-shop scheduling problem with finite time horizon takes into consideration set-ups (condition (3)), overtime (conditions (1) and (2)), buffer stocks and shortfall considerations (equation (4), condition (5), and objectives (6) and (9)). The formulation considers initial and final states of the inventory levels as zero or non-zero and machine status as busy or idle (equation (4) and condition (5)). It allows for batch size and number of batches to be variable (equation (4) and condition (5)) and it considers set-ups as explicit entities (condition (3)). Job priorities (objective (9)), multiple machines or machine banks (equation (1)), and multiple optimality criteria (objectives (6) through (10)) are also considered within this formulation.

4.4 Size of the Problem Formulation

The problem is to process a set of J jobs on a set of machines, M(p). Each job consists of a set of I(j) operations, (o(j,1) through o(j,I(j))), to be performed in sequence. Each operation o(j,i) requires the exclusive use of a particular machine for a duration, p(j,i), the processing time of the operation. To solve the problem, one must determine the number of machines, x(j,i,t), which should be processing operation i of job j at time interval t

in order to meet inventory requirements, d(j,i,t), minimize in-process inventory, Y(j,i,t), minimize set-ups, s(j,i,t) and minimize overtime, V(p,t).

The size of the problem formulation is given in terms of the number of decision variables and the number of constraints. We are given the job processing sequence for each job and the processing time per operation, p(j,i). We are also given the machine data, p, m(p), and M(p), the set-up times, f(j,i), and the maximum overtime allowed, W(p,t). Cost data in the form of CS(j,i,t), CV(p,t), CP(j,i,t) and CH(j,i,t) are provided as initial input data for the problem. Finally, the desired inventory levels, d(j,i,t), and the initial states of x(j,i,0) and Y(j,i,0) for machines and inventory are also provided.

The decision variables are the number of machines processing operation i of job j at time t, x(j,i,t), the number of machine set-ups, s(j,i,t), the in-process inventory, Y(j,i,t), and the overtime, V(p,t). Both x(j,i,t) and s(j,i,t) take on integer values, while Y(j,i,t) and V(p,t) are real valued. We have J total jobs, I(j) total operations per job, P total operation types, and T time periods. In order to simplify the computations for the number of decision variables and constraints, let us work with the special case where each job contains I operations and one operation is to be processed on each of the machines, I(j) = I for all j. From this, we compute the number of constraints and the number of decision variables.

The machine number constraints (condition (1)) are defined for all unique machine operations, p, and all time periods, t. Thus, we have P times T machine number constraints. The overtime availability constraints (condition (2)) are also defined for all p and all t and are of the number P times T. The set-up constraints (equation (3)), the in-process inventory definition (equation (4)), and the technological constraints (condition (5)) are defined for all jobs, all operations and all time periods. There are J times I times T constraints for each of these types of constraint. (Note: Since condition (5) is not defined for when i equals t equals 1, we have J times I times T minus J for this particular constraint.)

The overtime decision variables are defined for all p and all t. There are P times T of these decision variables. The set-up, in-process inventory, and the number of machines decision variables are defined for all jobs, for all operations and for all time periods. Thus, there are J times I times T set-up decision variables, J times I times T in-process inventory decision variables, and J times I times T machine number decision variables. It should be noted that the set-up decision variables are a function of the machine number decision variables and the in-process inventory decision variables are a function of both the set-up and machine number decision variables.

The total number of constraints is T times P times 2 plus T times J times I times 3 minus J. This sum is: $T (2 P + 3 J I) - J$. The total number of decision variables is T times P plus T times J times I times 3. This sum is: $T (P + 3 J I)$. There are of course non-negativity constraints on each of the decision variables, but these are not counted explicitly here. For a problem with 5 jobs, 5 unique operations per job (all jobs have the 5 operations in common) and 10 time periods, we have 10 (10 + 3x5x5) - 5 or 845 constraints (not counting the non-negativity constraints) and 10 (5 + 3x5x5) or 800 decision variables. For a 10 job, 10 (common) operations per job and 10 time period problem, we have 3190 constraints and

3100 decision variables. Since the size of the most general problem formulation is rather large, it may be worthwhile to consider an augmented formulation in order to reduce the problem size, e.g., ignore set-up decision variables and corresponding constraints to reduce the problem size.

The size of the problem is directly proportional to the number of time periods, T. If we choose a smaller time period and attempt to capture the dynamics of the scheduling environment, then T will generally be large. If we choose the time step to be larger, then the value of T should be smaller and the size of the problem will be reduced. By choosing a larger time step, however, we may not properly represent the dynamics of the real-time rescheduling problem. The choice of T will depend on the values for the set-up times and the job processing times. We wish to choose the smallest T such that we can accurately model the set-up times and job processing times.

5.0 The Simulation-Optimization Solution Strategy

The solution strategy for the problem formulation relies upon a synthesis of deterministic scheduling heuristics and discrete-event simulation. This simulation-optimization approach uses deterministic scheduling heuristics to select candidate schedule(s) and uses discrete-event simulation to provide quantitative measures for the robustness criteria of these candidate schedule(s), i.e., to provide measures for how well the schedule performs under a range of random disturbances.

Figure 3 is a schematic diagram of the conceptual framework for the real-time production scheduling decision aid. The first task to be completed when using the decision aid is to ready the input data and select the desired scheduling objective(s). Within the total set of scheduling objectives described in Section 4.3, the user is requested to choose the most important objective and/or rank the objectives in their order of importance. This is done in case a feasible schedule which is to meet all the objectives can not be generated, i.e., the schedule will be generated to meet the highest priority objectives.

The decision aid uses the rank-ordered objectives set and the scheduling input data base and generates an initial feasible schedule. This initial schedule is created from the deterministic solution to the mixed integer, multiple objective, linear programming problem. The solution to this problem requires the use of a heuristic solution procedure specifically tailored to mixed integer problem types. The decision aid analyzes the problem structure and chooses the "best" heuristic to be used to solve the problem and thereby provide an initial feasible schedule.

If an initial feasible schedule can not be generated by the decision aid, then the flow plan must be augmented (see Figure 1). Within this augmentation process, several alternative courses of action may be considered (Holstein; 1968). Alternative actions may include attempting another job priority ranking, cutting the lot size (i.e., breaking the job into two lots), trying an earlier start time (if tool, material, and so on are available), negotiating a later delivery date, and/or allowing the program to overload one or more work centers after agreeing with the manufacturing supervisors on an overtime plan or other capacity adjustment to handle the overload.

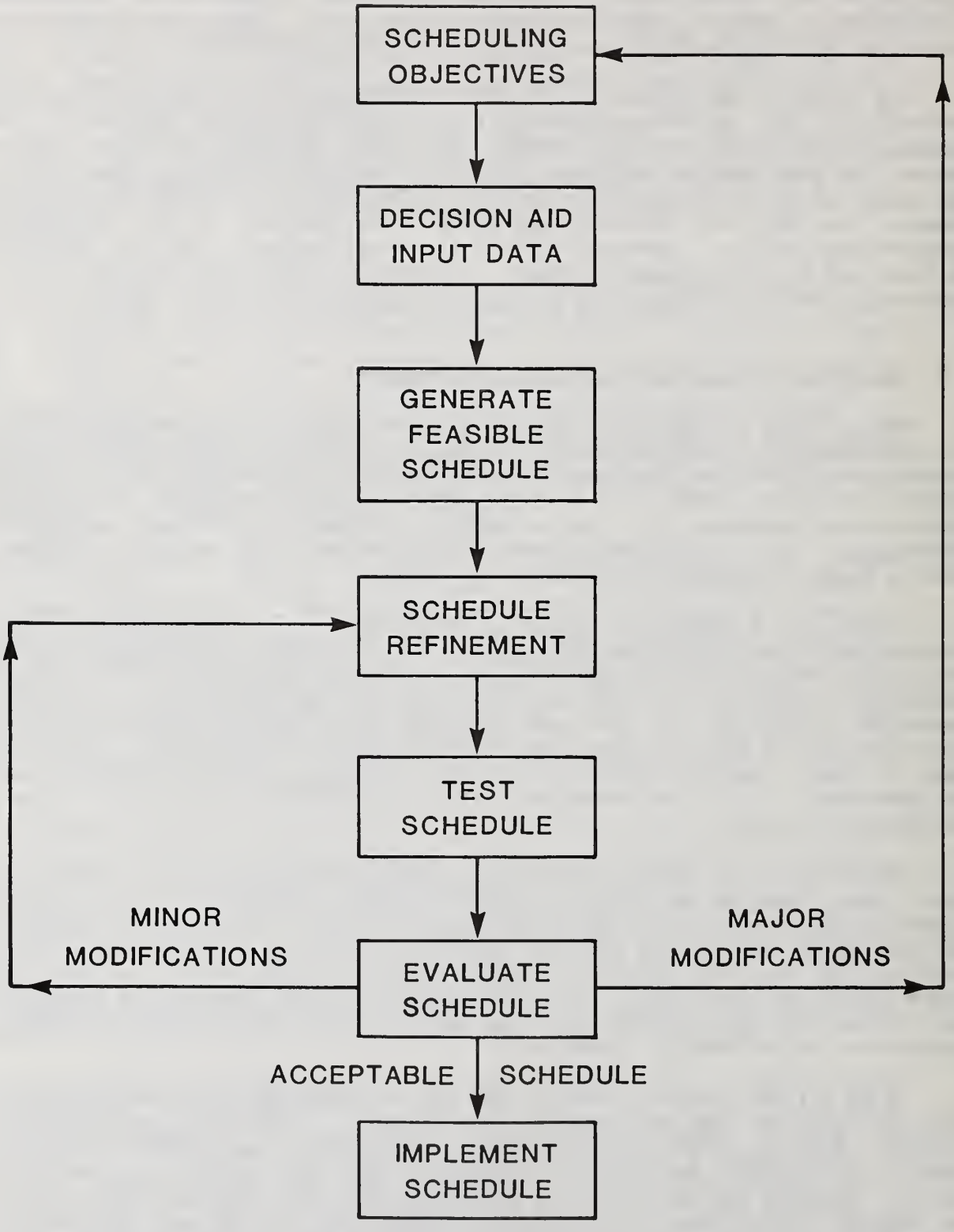


Figure 3: CONCEPTUAL FRAMEWORK FOR THE REAL-TIME PRODUCTION SCHEDULING DECISION AID

Continuing with Figure 3, we note that the initial feasible schedule serves as the "seed" for the schedule refinement step. In the schedule refinement step, the decision aid attempts to find a better schedule(s) by applying "optimizing" heuristic solution approaches to the mixed integer, multiple objective, linear programming problem. This solution refinement approach is similar to solution approaches which combine partial enumeration, heuristics, and neighborhood search strategies, i.e. combinatorial-optimization heuristics.

The refined schedule(s) is then tested with a simulation model which contains data on the actual manufacturing facility model for which the schedule is to be applied towards. The schedule is tested under two conditions within this dynamic simulation environment. First, the schedule is tested under normal facility operating conditions in order to determine how the randomness of the facility environment may affect the recommended schedule. The random elements to the first testing condition include stochastic job processing times and set-up times, and minor deviations in the desired inventory levels and demand levels.

In the second testing environment, the schedule is tested under "disruptive" facility operating conditions. The second test allows the scheduler to evaluate the schedule under disruption conditions such as machine failures, material unavailability, surges in demand, or other "what-if" type scenarios. These disruptive test conditions allow the scheduler to provide quantitative measures for the robustness and flexibility criteria of the schedule design process.

After the testing of the schedule in the simulation environment is completed, the scheduler may choose one of three options. If major modifications are required of the schedule, then the scheduler may wish to re-rank the order of the scheduling objectives and begin a new scheduling session with a revised objectives ordering. Under this option the scheduler is realizing that the initial rank order of objectives can not produce a desirable schedule. The second modification option allows the scheduler to perform minor modifications (or refinements) to the recommended schedule. Here the simulation evaluation results for the testing of the schedule are used within the decision aid to further refine the existing schedule, i.e., the simulation is used as a schedule design refinement tool. If the resulting evaluation criteria of the most recent schedule are acceptable, then the third option, the schedule implementation option, is chosen. If the schedule is implemented, then the input data base is updated to reflect the most current information on the (new) schedule in operation.

6.0 Summary and Future Research

We have proposed a multiple objective, mixed integer, linear programming formulation of the real-time scheduling problem. We now need to explore algorithmic solution procedures for solving this problem formulation and determine what heuristics and simplifications are possible. The "best" solution to the integer programming problem needs to be determined and then tested via simulation in a "real-time" environment. Our future research will concentrate on development of an algorithmic solution procedure and incorporating this procedure into a working prototype for the decision aid. Human factor issues and data base needs for the decision aid will also be

researched. It should be noted that, due to the complexity of the problem, it may be impossible for the working prototype to perform all of its functions in real-time.

ACKNOWLEDGEMENTS

This research was sponsored jointly by the Commonwealth of Virginia's Center for Innovative Technology and the General Electric Company. Special thanks is expressed towards Mr. Walter Trybula and Mr. Ricki Ingalls of General Electric's Electronics Automation Applications Center in Charlottesville, Virginia for their expert assistance and support.

REFERENCES

Abraham, C., Dietrich, B., Graves, S. and Maxwell, W., "A Research Agenda for Models to Plan and Schedule Manufacturing Systems", July, 1985.

Afentakis, P., "Simultaneous Lot Sizing and Sequencing for Multistage Production Systems", IIE Transactions, December, 1985, pp. 327-331.

Akella, R., Choong, Y. and Gershwin, S. B., "Performance of Hierarchical Production Scheduling Policy", IEEE Transactions on Components, Hybrids, and Manufacturing Technology, Vol. CHMT-7, No. 3, September, 1984, pp. 225-238.

Baker, C. T. and Dzielinski, B. P., "Simulation of a Simplified Job Shop", Management Science, Vol. 6, No. 3, April, 1960, pp. 311-323.

Bowman, E. H., "The Schedule-Sequencing Problem", Operations Research, Vol. 7, 1959, pp. 621-624.

Bulkin, M. H., Colley, J. L. and Steinhoff, Jr., H. W., "Load Forecasting, Priority Sequencing, and Simulation in a Job Shop Control System", Management Science, Vol. 13, No. 2, October, 1966.

Conway, R. W., Maxwell, W. L. and Miller, L. W., Theory of Scheduling, Addison-Wesley, Reading, Massachusetts, 1967.

Davis, L., "Job Shop Scheduling with Genetic Algorithms", Proceedings of an International Conference on Genetic Algorithms and Their Applications, Carnegie-Mellon University, Pittsburgh, Pennsylvania, July 24-26, 1985.

Fox, K. A., "MRP-II Providing a Natural Hub for Computer-Integrated Manufacturing System", Industrial Engineering, October, 1984, pp. 44-50.

Fox, M. S., Allen, B. P., Smith, S. F. and Strohm, G. A., ISIS: A Constraint-Directed Reasoning Approach to Job Shop Scheduling: System Summary, Carnegie-Mellon University, The Robotics Institute, CMU-RI-TR-83-8, June, 1983.

French, S., Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop, Wiley, New York, 1982.

Gere, W. S., "Heuristics in Job Shop Scheduling", Management Science, Vol. 13, No. 3, November, 1966, pp. 167-190.

Gershwin, S. B., Akella, R. and Choong, Y. F., "Short-term Production Scheduling of an Automated Manufacturing Facility", IBM Journal of Research and Development, Vol. 29, No. 4, July, 1985, pp. 392-400.

Gershwin, S. B., Hildebrant, R. R., Suri, R., and Mitter, S. K., "A Control Theorist's Perspective on Recent Trends in Manufacturing Systems", Presented at the 23rd IEEE Conference on Decision and Control, Las Vegas, Nevada, December, 1984.

Graves, S. C., "A Review of Production Scheduling", Operations Research, Vol. 29, No. 4, August, 1981, pp. 646-675.

Hax, A. C. and Meal, H. C., "Hierarchical Integration of Production Planning and Scheduling", Studies in the Management Sciences, Vol. 1: Logistics, edited by M. A. Geisler, North Holland-American Elsevier, 1975.

Herrod, R. A. and Papas, B., "Artificial Intelligence Moves Into Industrial and Process Control", The Industrial and Process Control Magazine, March, 1985, pp. 45-60.

Holstein, W. K., "Production Planning and Control Integrated", Harvard Business Review, May-June, 1968.

Hurrion, R. D., "An Investigation of Visual Interactive Simulation Methods Using the Job Shop Scheduling Problem", Journal of the Operational Research Society, Vol. 29, No. 11, 1978, pp. 1085-1093.

Jacobs, F. R., "OPT Uncovered: Many Production Planning and Scheduling Concepts Can Be Applied With or Without the Software", Industrial Engineering, October, 1984, pp. 32-41.

Karmarkar, U. S., Kekre, S., Kekre, S. and Freeman, S., "Lot-Sizing and Lead-Time Performance in a Manufacturing Cell", Interfaces, Vol. 15, No. 2, 1985, pp. 1-9.

Kivenko, K., Managing Work-In-Process Inventory, Marcel Dekker, Inc., New York, 1981.

Lasserre, J. B., Bes, C. and Roubellat, F., "The Stochastic Discrete Dynamic Lot Size Problem: An Open-Loop Solution", Operations Research, Vol. 33, No. 3, 1985, pp. 684-689.

McPherson, R. F. and White, Jr., K. P., "A Management Control Approach to the Manufacturing Planning and Scheduling Problem", Presented at the National Bureau of Standards Symposium on Real-Time Optimization in Automated Manufacturing Facilities, Gaithersburg, Maryland, January 21-22, 1986.

Mellor, P., "A Review of Job Shop Scheduling", Operational Research Quarterly, Vol. 17, No. 2, June, 1966, pp. 161-171.

Panwalkar, S. S. and Iskander W., "A Survey of Scheduling Rules", Operations Research, Vol. 25, No. 1, January-February, 1977, pp. 45-61.

Rinnooy Kan, A. H. G., Machine Scheduling Problems: Classification, Complexity and Computations, Martinus Nijhoff, The Hague, 39, 1976.

Ross, S. M., Applied Probability Models with Optimization Applications, Holden-Day, San Francisco, 1970.

Ross, S. M., Introduction to Stochastic Dynamic Programming, Academic Press, New York, 1983.

Schonberger, R. J., "Just-In-Time Production Systems: Replacing Complexity With Simplicity In Manufacturing Management", Industrial Engineering, October, 1984, pp. 52-63.

Shannon, R. E., "Artificial Intelligence and Simulation", 1984 Winter Simulation Conference Proceedings, edited by S. Sheppard, U. Pooch and D. Pegden, 1984.

Stecke, K. E. and Solberg, J. J., "Loading and Control Policies for a Flexible Manufacturing System", International Journal of Production Research, Vol. 19, No. 5, 1980, pp. 481-490.

Stecke, K. E. and Solberg, J. J., "The Optimality of Unbalanced Workloads and Machine Group Sizes in Closed Queueing Networks of Multiserver Queues", Operations Research, Vol. 33, No. 4, 1985, pp. 882-910.

Steinberg, E. and Napier, H. A., "Optimal Multi-Level Lot Sizing for Requirements Planning Systems", Management Science, Vol. 26, No. 12, December, 1980, pp. 1258-1263.

Trybula, W. J. and Ingalls, R. G., "Simulation of Hybrid Automation", General Electric Company, Electronics Automation Application Center, Charlottesville, Virginia, 1985.

White, Jr., K. P. and Rogers, R. V., "Formulation of the Job Shop Scheduling Problem As An LP With Restricted Basis", University of Virginia, Charlottesville, Virginia, May, 1985.

Winters, P. R., "Constrained Inventory Rules for Production Smoothing", Management Science, Vol. 8, 1962, pp. 470-481.

AN OPTIMIZED KANBAN SYSTEM

FOR A CUSTOM DOOR MANUFACTURER

by

Wayne J. Davis
Department of General Engineering
University of Illinois at Urbana
104 S. Mathews
Urbana, Illinois, 61801, U.S.A.

INTRODUCTION

This paper considers an actual case study of a custom door manufacturer. The manufacturer experiences a wide variety of production sequences arising from the differing types and styles of doors among the received orders. However, in reviewing past orders, it was found that three primary production sequences accounted for nearly 75% of the doors ordered while two basic production sequences accounted for nearly 95% of the frames. This paper will limit its consideration to these five production sequences. Door types associated with each production sequence will be referred to as Door Type I, II, and III, while the frame types will be denoted as Frame Type I and II. The sequence of production steps used in the manufacturing of each door and frame type is given in Table 1 and diagramed in Figure 1. As presented, there is a maximum of ten primary production workstations to be considered, with two additional workstations serving for the staging and shipping of orders.

The manufacturing environment falls within two extreme production situations: the pure flow shop and the pure job shop. In the pure flow shop situation, orders flow in a completely deterministic manner through each subsequent production process. However, in the pure job shop, an order flows from one workstation to another in an apparently random fashion. Both situations are idealizations of reality and seldom exist. Looking at Figure 1, the production situation is nearly a flow shop. That is, if a production step is employed in the manufacturing of a given order, it must be employed in a specific chronological order. The major distinction from the flow shop environment is that for a given door or frame type, certain production steps may be omitted.

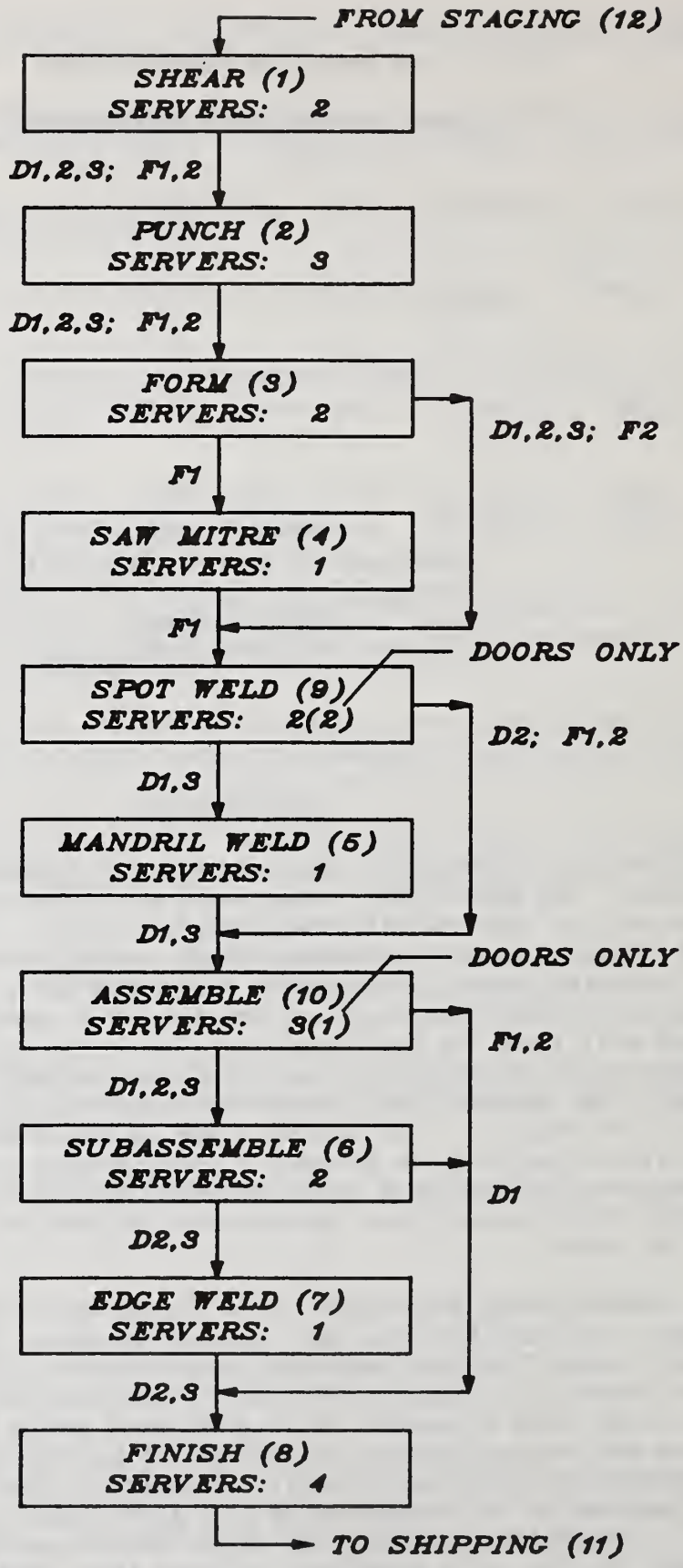


Figure 1 -- Production Flow Diagram for Door Manufacturer

PRODUCTION STATION (STATION NUMBER)	DOOR TYPE			FRAME TYPE	
	1	2	3	1	2
SHEAR (1)	X	X	X	X	X
PUNCH (2)	X	X	X	X	X
FORM (3)	X	X	X	X	X
SAW MITRE (4)				X	
SPOTWELD (9)	X	X	X	X	X
MANDRIL WELD (5)	X		X		
ASSEMBLE (10)	X	X	X	X	X
SUBASSEMBLE (6)	X	X	X		
EDGE WELD (7)		X	X		
FINISH (8)	X	X	X	X	X

ADDITIONAL STATIONS:
STATION 11: SHIPPING
STATION 12: STAGING

Table 1--Production Flow chart

In the past, the company has experienced considerable problems with production scheduling. To maximize flow through the factory, the company has adopted the strategy of releasing orders at the first availability of the shearing workstation. Over time, this policy has resulted in severe congestion of the shop floor primarily because the production times at the workstations are not balanced. Specifically, certain workstations such as spotwelding require significantly more processing time than the other workstations. The orders collect in queues at the critical workstations causing congestion which further increases production times at these stations. Ultimately, orders were being misplaced, and the time that orders spent on the production floor was continuing to increase. To meet promised delivery dates, the company was constantly expediting a given order. Finally, the work-in-progress inventory was becoming a major production cost. It was clear that the policy of earliest possible order release was not functioning as desired.

Early analysis for improved production scheduling sought to define an explicit production control law that would continuously monitor the system and determine when and which of the queued orders would be released to the shop floor next. The definition of this control law would rely primarily upon backward scheduling techniques. Although the definition of an active control law is certainly possible, it was believed that the assignment of optimal parameters within the model would be difficult. Furthermore, the capability to continuously monitor the total production system was not easily implemented and perhaps totally infeasible for the small manufacturer.

Recently the Kanban approach has been employed by Japanese manufacturers and others for production control. Briefly, the objective of the Kanban's approach is to limit the number of orders that are either being processed or have been processed by a given workstation, but have not completed the next production step. As displayed by Figure 2, the Kanban approach to be adopted

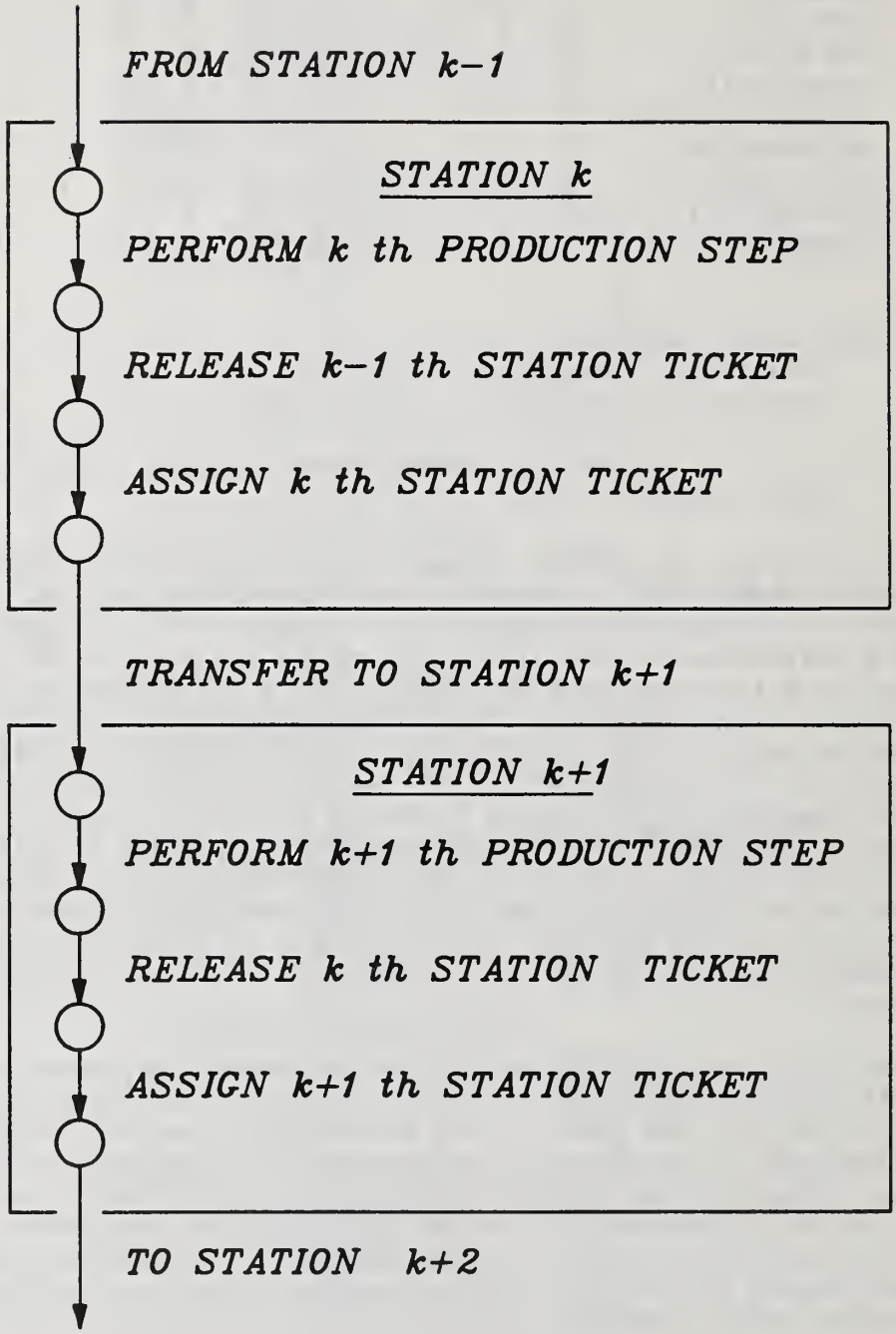


Figure 2 -- Proposed Policy for Assignment of Kanban Cards

in this study will not allow the server or the order to be released at a given workstation until that station's card can be assigned to the processed order. That is, the server will remain idle until a Kanban card can be assigned. On the other hand, an issued card must remain with the order until the subsequent production step is completed. By limiting the number of cards assigned to a given station, the previously stated objective for the Kanban approach is implemented.

An alternative approach would issue the card before production could begin at the station. The latter configuration will typically require that more cards be allocated to each workstation. The former approach was adopted in this study to increase the probability that an order would be ready for transfer to the next station whenever a ticket became available. It is not believed that either configuration when properly optimized will generate significantly better performance than the other. However, this point could be tested.

The Kanban approach has been previously employed in production environments representing a nearly pure flow shop with balanced production times at each station. This type of production environment is one of the easiest to control as the flows are predictable, and there is no inherent tendency for orders to collect at any specific station. When the production line is functioning properly, the Kanban approach has little or no influence upon the production. On the other hand, when excursions from the ideal or equilibrium production situation do occur, the Kanban procedure tends to limit the system's transgressions. Finally, since the line is presumed to be balanced, the number of cards issued to each station is usually a constant.

Application of the Kanban approach to the door manufacturing problem represents two principal departures from the production environment cited above. First, a pure flow shop does not exist, and therefore the flows from station to station are not deterministic. Second, the production times at each station are not balanced, and there is a tendency for orders to collect at critical workstations. The immediate consequence of the these observations is that no ideal steady state production configuration exists. It can be anticipated that the Kanban procedure will continuously be constraining the system, hence exerting a more active influence in production control. Finally, since the production times are not balanced, the number of cards allocated to each station will differ. This paper will seek to determine the optimum number of Kanban cards to be assigned to each station.

THE SIMULATION MODEL

Before the optimization could be undertaken, a simulation model for the door manufacturer was developed using the simulation SIMAN [3]. SIMAN was chosen for three primary reasons:

1) The simulation language was available for the personal computer, providing a nearly unlimited computational resource.

2) The language provided a means for modeling the workstation configuration of the door manufacturing environment.

3) The language also contained mechanisms for modeling the transport of materials between workstations.

Using SIMAN, a generic workstation was defined for stations 1 through 8 as denoted in Figure 1. For these stations, only the service times and the number of servers at each station varied. For stations 9 and 10, special coding was required to permit the inclusion of servers who could process doors only. Finally, stations 11 and 12 were specifically coded for their respective shipping and staging functions.

SIMAN also allowed the inclusion of material handling carts to permit the modeling of the transport function between the workstations. Pertaining to this transport function, a pallet of either doors or frames was defined as the maximum number of units that can be transported at one time by the transporter. To provide variability in order size, it was assumed that a door order would require one, two, or three pallets with a probability of 0.5, 0.3, and 0.2, respectively. Associated with each door order would be a corresponding order for frames requiring the same number of pallets. Partial pallets were not considered explicitly, but rather considered implicitly by allowing variations in processing times.

Programming the system in SIMAN was relatively straightforward except for a few particular features included in the model. First, each arriving order was split into two distinct door and frame orders which were subsequently treated as separate entities during the production processes. The door and frame orders were released to the production floor on a first-in-first-out basis. At a given workstation, it was required that all pallets to an order of either doors or frames be finished before the Kanban card would be assigned and the order could be transported to the next production process. However, once the first pallet for a given order arrived at a station, processing could commence at the first availability of server. Furthermore, the same server was required to process all pallets associated with a given order. When the processing of all pallets was complete, the Kanban card to the preceding station was then released. At the shipping station, a completed door or frame order remained until its counterpart was also ready for shipping. To provide additional control upon the total orders on the production floor, another card was assigned to each door or frame order upon submission to the shop floor. The ALL cards remained with the individual door and frame orders until both the door and frame order comprising a total order were ready for shipment.

OPTIMIZATION ON THE RESPONSE SURFACE

Using the Kanban approach, the goal is to establish an optimal allotment of cards at each of the twelve workstations as well as the allotment of ALL cards which determine the maximum number of orders permitted on the floor at any time. Determining this optimal allocation is difficult. Unlike the optimization of a function which can be mathematically specified, the functional form of the criteria to be considered in this optimization is not explicitly known. Instead, the criteria must be evaluated experimentally through simulation. This evaluating procedure is further complicated by the stochastic elements in the observed system. The procedures which will be employed in the evaluation have been collectively termed response surface methodology (RSM) (see Law and Kelton [2]).

Unfortunately, the techniques of RSM are not immediately applicable to this study. RSM presumes that the function or response being studied is continuous on a defined interval. For this study, the investigated responses

will be defined at discrete or integer values only as fractional allocations of cards have no meaning. Nevertheless, an attempt will be made using the RSM procedures to approximate the derivatives for the criteria functions at a given point. The quality of any approximation will be affected by the consideration of discrete variables only. However, these approximations may provide insight into potential reallocation of cards to improve the selected responses to be optimized. Also these approximations may provide verification of a point as a local optimum. It is not believed that global optimality can be demonstrated, however, due to the integer nature of the problem.

DETERMINING AN INITIAL CONFIGURATION

As noted above, there are thirteen potential factors (the number of cards at each station and the number of ALL cards) to be considered in the optimization. The first step in the optimization is to establish an initial assignment of values to these factors. The following procedure was employed. First, the mean interarrival time for orders was set to three hours, and a simulation was performed with a known excess of cards at each station. The attempt here was to determine the free response of the system without saturation effects due to excessive order arrivals or constraining limitations due to the availability of cards. The number of cards at each station was then successively reduced until further decreases would effect the system response. The process is significantly simplified by using the output statistics generated by SIMAN for the utilization of cards at each station. Since only an initial configuration was being sought for the optimization, the procedure did not need to be exact. The selected initial configuration is given in Table 2.

STATION	1	2	3	4	5	6	7	8	9	10	11	12	ALL
CARDS	4	4	4	2	3	4	1	10	7	7	7	3	25

Table 2--Initial Allocation of Cards for the Simulation Study

The next step was to establish a mean interarrival time for orders that would nearly saturate the system at this initial configuration. The saturation requirement was chosen to enhance the effects derived in varying system parameters. Initially two simulations were made with mean interarrival times of 2.0 and 2.5 hours, respectively. The results with respect to the three potential system responses -- the average time that an order is in the system, the average time that the order is on the production floor and the total time to produce one thousand orders -- are given in Table 3.

	MEAN INTERARRIVAL TIME (HOURS)		
	2.0	2.5	2.25
TIME IN SYSTEM	159.6	32.3	47.6
TIME ON FLOOR	32.9	29.1	32.3
TIME FOR 1000 ORDERS	2280	2586	2315

Table 3--Measured Responses for Mean Interarrival Times

The interarrival time of 2.0 hours saturated the system as evidenced by the significant increase in the average time an order resides in the system. Using the available information on the total time to process 1000 orders, the minimum average interdeparture time at saturation can be computed as 2.28 (=2280/1000) hours. Since arrivals are occurring on average every 2.0 hours, the system is obviously saturated. An average interarrival time of 2.5 hours does not appear to saturate the system. For this case, the mean interdeparture time is 2.59 (=2586/1000) hours.

Another simulation was then made with a mean interarrival time of 2.25 hours for which the responses are also given in Table 3. The growth for the TIME IN SYSTEM response displays some saturation in the system. The mean interdeparture time for this case is 2.32 (=2315/1000) hours. Since the differences between the mean interdeparture and interarrival times for the 2.25 and 2.5 hour cases are approximately the same, the saturation does not appear to be severe. Hence, 2.25 hours will be used as the mean interarrival time for all subsequent simulation runs.

Finally, to provide a basis for the comparison of the effectiveness of the Kanban control strategy, another simulation was run with an excess of cards at every station. The system responses gave TIME IN SYSTEM (TIS) = 66.2 hours, TIME ON FLOOR (TOF) = 66.1 hours, and TIME FOR 1000 ORDERS = 2316 hours. Returning to Table 3, the effectiveness of the Kanban approach becomes immediately evident. Using the initial allotment of cards (and the 2.25 hour case), the average time in the system is reduced by 28% [=(66.2-47.6)/66.2] and the average time on the floor by 51% [=(66.1-32.3)/66.1]. The time required to process 1000 orders is nearly identical in both cases.

Although the present reductions in both the TIS and TOF responses are substantial, the question remains whether the system can be further optimized. A particular concern is whether both indices can continue to be simultaneously reduced or must one index be compromised to effect a reduction in the other. The former case is the ideal situation. However, if the second case occurs, it is essential that the nature of the compromise be quantified.

ITERATION ONE

Using the concepts of RSM, ideally an approximation to the response would be developed at each iteration which would then be used to define a new point that would improve the responses on the next iteration. One accepted approach is to use the resulting approximation to subsequently define an approximation to the gradient for the response which directs the generation of an improved design configuration for the system. The success of this approach is necessarily dependent upon the ability of the analyst to develop an adequate approximation to the function at the current design point. This capability will be discussed later.

From the outset, thirteen factors were viewed as too many factors for consideration. Using the preliminary simulation results, six factors were dropped from consideration. The number of tickets (or factors) associated with stations 1, 4, 5, and 7 were eliminated because the initialization has already assigned a minimal number of tickets to these stations, while the utilization statistics for these factors generated from the simulations do not justify major changes in their assigned values. Both the station 12 card and

the ALL card are issued at the staging station Therefore the station 12 card which has already been assigned a minimal value is somewhat redundant. The station 11 card (or shipping) is not really needed as there are no subsequent processes. Finally, the remaining seven factors form an ideal number for employing fractional factorial design to specify configurations to be simulated.

The initial assignment of factors to the cards at a specific workstation is given in Table 4. Also given for each selected factor are a high value x_i^+, and a low value, x_i^-, which are symmetrically placed about the initial setting, x_i^0. The techniques of experimental factorial design will be employed to define the simulation runs (see Box, Hunter, and Hunter [1] or Law and Kelton [2]). Using the responses from these simulation runs, estimates of the main effects for each factor, denoted by <i> for i=1,...7, can be made. Letting y denote the desired response to be approximated, the $\underline{X}^0$ is the vector of initial setting of the factors X_i is the normalized factor main effect <i> represents a statistical evaluation of $2 \frac{\partial y}{\partial X_i} \Big|_{\underline{X}^0}$ where which assumes values on an interval [-1,1] when x_i assumes values on the interval $[x_i^-, x_i^+]$ or

$$X_i = [x_i - (x_i^+ + x_i^-)/2] \; / \; [(x_i^+ - x_i^-)/2] \quad (i=1,\ldots,7). \qquad (1)$$

A linear approximation to the response y can then be generated as

$$y\,(X_1,\ldots,X_7) \simeq \bar{y} + \sum_{i=1}^{7} \frac{\langle i \rangle}{2} X_i \qquad (2)$$

where $\bar{y}$ is the computed average response over all simulation runs. The quality of this approximation depends upon both the size of the higher order effects and the size of the experimental grid for defining the simulations. Of particular interest are the two-factor effects, <ij> for i,j=1,...,7 and $i \neq j$, which statistically evaluate $2 \frac{\partial^2 y}{\partial X_i \partial x_j} \Big|_{\underline{X}^0}$. These effects represent the anticipated changes in the response derived from modifying two factors simultaneously. Three- and higher-factor effects can also be computed.

To estimate all potential effects, 2^7 (=128) simulation runs would be required. This number of runs is too large considering the rather coarse grid pattern with which the initial design was specified. Therefore, a fractional factorial design was chosen [1]. Specifically, a 2^{7-4} design was generated by using the design generators I=124=135=236=1237. In Table 5, the factor settings for each simulation run are given. A "+" indicates that x_i would be set to x_i^+, while "-" indicates setting x_i to x_i^-. Also included in the table are the measured averages for both the TIS and TOF responses on each associated simulation run.

FACTORS		(1)	(2)			(3)		(4)	(4)	(6)			(7)
STATIONS	1	2	3	4	5	6	7	8	9	10	11	12	ALL
x_i^0	3	4	4	2	3	4	1	10	7	7	7	3	25
x_i^+		5	5			5		12	9	9			28
x_i^-		3	3			3		8	5	5			22

Table 4--Factorial Specifications for Iteration One

RUN	1	2	3	12 4	13 5	23 6	123 7	TIS (hrs)	TOF (hrs)
1	-	-	-	+	+	+	-	62.5	30.4
2	+	-	-	-	-	+	+	43.1	34.2
3	-	+	-	-	+	-	+	47.4	34.3
4	+	+	-	+	-	-	-	69.0	31.0
5	-	-	+	+	-	-	+	45.9	34.7
6	+	-	+	-	+	-	-	64.5	31.3
7	-	+	+	-	-	+	-	54.5	29.5
8	+	+	+	+	+	+	+	39.4	32.6

Table 5--Simulated Configurations and Computed Responses for Iteration One

From the collected data, the behavior of both responses appears complicated. Attempts at defining an appropriate linear approximation were made, but the associated errors were too large to allow substantial reliance upon the gradients derived from the approximation. This approximation was certainly compromised by the fact that each main effect was confounded with two factor effects. Eight additional simulation runs would be required to remove this confounding of main effects with two factor effects. Thus, the gradient could not be employed to determine the base point for the next experimental design. Rather, the configuration associated with simulation run 7 yielded desirable reductions of both the TIS and TOF responses. Therefore, this configuration will be subjected to additional investigation during subsequent iterations.

ITERATION TWO

On iteration one, fractional factorial design was employed to generate approximate information pertaining to all effects using a small number of simulation runs. On iteration two, the factorial design will be abandoned to better illustrate the behavior of the main effects for the responses. Specifically, using the initial factor settings arising from run seven on iteration one given in Table 6, three sets of simulation runs were made. On the first and second set of runs, each of the factors was increased individually by one and two, respectively. On the third set of simulation

runs, each factor was decreased by one. Thus, a total of twenty simulation runs were made. (The simulation run increasing factor seven by two was not made.)

FACTORS	(1)	(2)			(3)			(4)	(5)	(6)			(7)
STATIONS	1	2	3	4	5	6	7	8	9	10	11	12	ALL
x_i^0	1	3	5	2	3	5	1	8	5	9	7	3	22

Table 6--Initial Factorial Setting for Iteration Two

The results of the simulation runs are given in Table 7. The results truly depict the complexity of the responses. For example, if factor two is increased by one, the average TIS drops from 54.5 to 44.6 hours, while increasing by another additional unit increases the average TIS to 46.5 hours. On the other hand, decreasing factor two by one unit again decreases the average TIS from 54.5 to 46.8 hours. Thus a relative maximum and minimum for the TIS response are demonstrated for factor two over an interval of three units. Since only discrete values for factor two are relevant, developing an approximation with respect to this factor appears impossible.

FACTOR (RUN)	TIS (hrs)			TOF (hrs)		
	+1	+2	-1	+1	+2	-1
1	52.0	44.2	57.9	29.6	28.5	30.5
2	44.6	46.5	46.8	29.5	30.3	30.0
3	54.5	54.5	48.5	29.5	29.5	29.9
4	54.5	54.5	54.5	29.5	29.5	29.5
5	53.0	48.6	52.9	29.7	28.6	30.6
6	54.5	54.5	54.5	29.5	29.5	29.5
7	61.5	----	79.7	32.1	----	30.4

Table 7--Simulation Results for Iteration Two

Factor three also demonstrates an interesting behavior in that increasing factor three causes no changes to either the TIS or TOF response. Decreasing factor three, however, causes the average TIS to decrease while the average TOF increases. Apparently, the number of cards at station 6 are initially imposing no constraint upon the system. Thus, adding extra cards does not change system response. However, when the number of cards is decreased, the system response is affected. Looking at the responses for factor three, it is also important to note that compromise among the responses may be required for all factors, not just factor seven which was the case in iteration one.

On iteration one, <4> appeared to be the second most significant. However, on iteration two, it appears that within the investigated range of values, factor four displays no effect upon either response. Two possibilities emerge: either the estimate for <4> on iteration one is

significantly in error or the significant factors can change depending on the current values assigned to the other factors. The latter case appears to be more likely. Factor six also appears to exhibit no influence upon the responses.

For factor seven, apparently a relative minimum has been located. This is especially important since factor seven controls the total number of orders allowed in the system at any time. Finally, factor one gives the most useful changes in both responses. Increasing or decreasing factor one by one unit results in changes to the average TIS response, which indicates a consistent decrease in the TIS with an increase in factor one. In fact, this was the observation that initiated the simulating of the +2 cases. When factor one was increased by two units, significant reductions occurred in both the TIS and TOF responses. As noted above, increasing factor two by one also gave desirable reductions. To test if the effects could be superimposed, a simulation run was made where factors one and two were increased by two and one, respectively. It resulted in an average TIS of 54.7 hours and an average TOF of 28.9 hours. Clearly, increasing only factor one by two yielded superior results.

Increasing factor five by two also resulted in desirable reductions to both responses. However, increasing both factor one and five by two units each gave an average TIS of 54.4 hours and average TOF of 30.0 hours. Therefore, increasing factor one by 2 yielded the best results on iteration two.

ITERATION THREE

The initial point for iteration three is given in Table 8. As shown in the table, three factors were dropped from further consideration, leaving only four. Iteration two showed that factors four and six were insignificant to the responses. For factor seven, a local minimum was located at the value of 22. On this iteration, a 2^{4-1} fractional factorial analysis was performed, requiring eight simulation runs. The configurations for the simulation runs are given in Table 9 with the associated results for the responses. In the table, a "+" implies increasing the factor by one while a "-" implies decreasing the factor by one. In every case, the measured responses are greater than the average TIS of 44.2 hours and average TOF of 28.5 hours corresponding to the initial point of the search. At this point, the initial point given in Table 8 will be treated as a local optima.

As a point of interest, the estimates of the average and main effects were made using the responses listed in Table 9. These estimates given in Table 10 are free of any confounding with two factor effects. It is first noted that estimated averages significantly overestimate the simulated average responses of 44.2 and 28.5 hours, respectively, at the base point for the experimental design. Again this is consistent with the base point being a local minimum. For the TOF response the main effects appear to be minimal. This fact is also true for the effects <3> and <4> on the TIS response. The effects <1> and <2> do appear significant. These effects would predict that reduction in TIS response could be made by decreasing either factor 1 or 2. Since both these situations have been simulated and resulted in an increase of the TIS response, the accuracy of the estimates is certainly questionable. These observations are especially important because we can not make estimates

of the compromise among the responses. Thus, a crucial consideration in multicriteria decision-making must be neglected. Due to the discrete nature of the factors better estimates of the effects appear impossible. Nevertheless, the final reductions given do appear to be significant, reflecting a 33% = (66.1-44.2)/66.1 reduction in the TIS response and a 57% = (66.1-28.5)/66.1 reduction in the TOF response.

NEW FACTOR	(1)	(2)			(3)			(4)					
OLD FACTOR	(1)	(2)			(3)			(4)	(5)	(6)			(7)
STATION	1	2	3	4	5	6	7	8	9	10	11	12	ALL
X_i^0	3	5	5	2	3	5	1	8	5	9	7	3	22

Table 8--Initial Factorial Settings for Iteration Three

RUN	EFFECTS 1	2	3	123= 4	TIS (hr)	TOF (hr)
1	-	-	-	-	47.9	29.5
2	+	-	-	+	49.9	30.1
3	-	+	-	+	48.4	29.9
4	+	+	-	-	53.0	29.8
5	-	-	+	+	46.8	29.6
6	+	-	+	-	46.5	29.6
7	-	+	+	-	51.5	29.7
8	+	+	+	+	55.3	29.7

Table 9--Factor Specification and Simulation Results for Iteration 3

COMPUTED EFFECTS	FOR TIS (HRS)	FOR TOF (HRS)
AVERAGE	49.90	29.70
<1>	.63	.03
<2>	1.07	.01
<3>	.05	-.04
<4>	.09	.04

Table 10--Estimated Main Effects for Iteration 3

CONCLUSION

The optimal solution obtained in the previous section does not represent an optimal configuration for the overall system. First, global optimality of the ascertained configuration, even with respect to the seven investigated factors, cannot be demonstrated. On the other hand, the improvements gained through the optimization procedure are self-evident. Second, system response with respect to the previously eliminated factors should be investigated. Here it is suggested that the approach of iteration two, perturbing each factor singularly, be adopted. If an improved configuration is defined in this process, then subsequent fine tuning of the seven analyzed factors may be required.

Finally, in the presented optimization, the structural configuration and assignment of servers to each station were taken as a given. An overall optimization of the system would also investigate the validity of this assumption. Particularly, the assignment of servers to each station as well as the number of transporters should be investigated. Fortunately, the simulations' outputs already provide substantial information pertaining to both server and transporter utilization. This information can be employed in determining an improved assignment which would be subsequently further optimized.

Although the above optimization tasks still remain, it is believed that several conclusions can be drawn from the presented results. First the conceptual basis for the Kanban approach is applicable to manufacturing systems which are not pure flow shops with balanced production times at each station. That is, the Kanban approach can be applied in a role of active production control beyond its typically implied function of system stabilization. Second, the optimization of a stochastic response on a discrete decision space has been explored. The complexity of binding and non-binding constraints, as well as the the changing configuration of critical factors, has been demonstrated. Finally, the necessity of compromise among conflicting system responses has been discussed. These, it has been shown that optimizing a manufacturing system is complex and usually not straightforward. Intuitions and current methodologies are not always relevant.

The optimization presented in this paper is, of course, off-line. The author believes that this off-line optimization is a crucial initial step toward an effective on-line real-time optimization. As stated in the introduction, initially an on-line control law was sought to govern when and which available orders were released to the shop floor. Although this study adopted the optimization of a Kanban configuration, this approach does not preclude the later inclusion of a optimization of which order should be released. Indeed, working within the Kanban system, considerable latitude still remains on when the order can be released. The adopted Kanban configuration, on the otherhand, provides the system with an improved response before such an optimal control law is sought. At a minimum the system has been stabilized. Current research will seek to the additional benefits that can be gained by the addition of real time optimization for the release of orders. It will also attempt to discover any sacrifices derived from the inclusion of the Kanban configuration over configuring the system will a real-time optimization only. In this manner, the interplay between off-line and real-time optimization can be investigated.

REFERENCES

(1) Box, G. E. P., W. G. Hunter, and J. S. Hunter, 1978, **Statistics for Experimenters: An Introduction to Design, Data Analysis, and Model Building** (New York: John Wiley & Sons, Inc.).

(2) Law, A. M. and D. Kelton, 1982, **Simulation Modelling and Analysis** (New York: McGraw-Hill Book Company).

(3) Pegden, C. P., 1985, **Introduction to SIMAN**, (State College, Pennsylvania: Systems Modeling Corporation).

A LINEAR PROGRAMMING APPROACH TO NUMERICALLY CONTROLLED FACE MILLING

Aseem Chandawarkar
Department of Industrial Engineering and Operations Research
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061

Richard A. Wysk
Department of Industrial and Management Systems Engineering
The Pennsylvania State University
University Park, PA 16802

1.0 Introduction

Face milling of a part involves moving the cutter relative to the part in such a manner that the entire surface is machined. More often than not, face milling involves making multiple parallel passes on each face to be machined so that the entire face is covered as shown in Figure 1. This type of milling is frequently referred to as staircase milling. A face may also be machined by starting along the periphery of the face and progressively moving inward in a spiral-like fashion until the entire face is covered. This method, called window-frame milling is illustrated in Figure 2.

In either case, the cutter path needs to be specified in terms of absolute or incremental dimensions of each path segment for it to be executed on a numerically controlled (NC) machine. The method of representing the part geometry significantly influences the determination of the cutter path. The Initial Graphics Exchange Specification (IGES) is becoming increasingly popular as a standard that assures communication compatibility between different CAD/CAM systems [TElCH85]. IGES uses a hierarchical structure where each geometric entity is specified by geometric entities of one less order (e.g., a solid is specified by its faces, a face by its edges and an edge by its end-point vertices). While such a representation provides explicitly all geometric features of the part, it is quite cumbersome to use, as is, in the determination of the cutter path. For each face to be machined it is first necessary to determine the "width" of the face (Figure 3) and divide it into the number of passes. It is, then, necessary to determine the edges between which each pass lies. This exercise requires that a number of sets of simultaneous equations be solved.

The computational effort can be considerably alleviated by employing linear programming (LP) to represent the part if the part is a convex polyhedron. A nonconvex polyhedron can be represented as a union of convex

polyhedra. The LP representation of a convex polyhedron and its ramifications in NC face milling are discussed in subsequent sections.

2.0 A Linear Programming Model of a Polyhedron

It is a well-known result in linear programming that the feasible region of a linear program is a convex polyhedron [TAHA76]. We restrict ourselves to three-dimensional (3-D) space with a Cartesian coordinate system consisting of x, y and z axes. We shall now develop a set of constraints such that the feasible region represents the polyhedron.

Any plane in 3-D divides the entire space into two "half-spaces". If a plane $A_i x + B_i y + C_i z = D_i$ is constructed so as to contain face i of the polyhedron, the convexity assumption about the polyhedron helps us conclude that the polyhedron lies entirely in one of the half-spaces created by the plane. Denoting the half-space containing the polyhedron and created by a plane containing face i by S_i, the polyhedron is completely defined by the intersection of half-spaces S_i for all faces i. Thus, without loss of generality, we can conclude that the following set of inequalities completely defines the polyhedron:

$$\begin{aligned} A_1x + B_1y + C_1z &\leq D_1 \\ A_2x + B_2y + C_2z &\leq D_2 \\ &\vdots \\ A_mx + B_my + C_mz &\leq D_m \end{aligned} \qquad (1)$$

The constants A_i, B_i, C_i and D_i can be determined by obtaining any three non-colinear points $P_{1i}(x_{1i},y_{1i},z_{1i})$, $P_{2i}(x_{2i},y_{2i},z_{2i})$ and $P_{3i}(x_{3i},y_{3i},z_{3i})$ in each plane i and a point $P_{in}(x_{in},y_{in},z_{in})$ lying strictly inside the polyhedron. The equation of the plane containing face i can be written as:

$$\begin{vmatrix} x & y & z \\ x_{1i}-x_{2i} & y_{1i}-y_{2i} & z_{1i}-z_{2i} \\ x_{2i}-x_{3i} & y_{2i}-y_{3i} & z_{2i}-z_{3i} \end{vmatrix} = \begin{vmatrix} x_{1i} & y_{1i} & z_{1i} \\ x_{1i}-x_{2i} & y_{1i}-y_{2i} & z_{1i}-z_{2i} \\ x_{2i}-x_{3i} & y_{2i}-y_{3i} & z_{2i}-z_{3i} \end{vmatrix} \qquad (2)$$

Comparing the equations (1) and (2), we get

$$A_i = \begin{vmatrix} y_{1i}-y_{2i} & z_{1i}-z_{2i} \\ y_{2i}-y_{3i} & z_{2i}-z_{3i} \end{vmatrix}, \quad B_i = \begin{vmatrix} z_{1i}-z_{2i} & x_{1i}-x_{2i} \\ z_{2i}-z_{3i} & x_{2i}-x_{3i} \end{vmatrix},$$

$$C_i = \begin{vmatrix} x_{1i}-x_{2i} & y_{1i}-y_{2i} \\ x_{2i}-x_{3i} & y_{2i}-y_{3i} \end{vmatrix}, \quad D_i = \begin{vmatrix} x_{1i} & y_{1i} & z_{1i} \\ z_{1i}-z_{2i} & x_{1i}-x_{2i} & z_{1i}-z_{2i} \\ z_{2i}-z_{3i} & x_{2i}-x_{3i} & z_{2i}-z_{3i} \end{vmatrix} \qquad (4)$$

Also,

$$A_i x + B_i y + C_i z \leq (\text{or } \geq)\ D_i \tag{5}$$

according as

$$A_i x_{in} + B_i y_{in} + C_i z_{in} < (\text{or } >)\ D_i$$

Non-colinearity of P_{1i}, P_{2i} and P_{3i} is essential for the identification of A_i, B_i, C_i and D_i and the non-coplanarity of P_{1i}, P_{2i}, P_{3i} and P_{in} is essential for the determination of the inequality sign in (5). The non-colinearity and non-coplanarity requirements can be tested using the following conditions:

$$\begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ x_{1i}-x_{2i} & y_{1i}-y_{2i} & z_{1i}-z_{2i} \\ x_{2i}-x_{3i} & y_{2i}-y_{3i} & z_{2i}-z_{3i} \end{vmatrix} \neq \vec{0} \tag{6}$$

and

$$\begin{vmatrix} x_{1i}-x_{in} & y_{1i}-y_{in} & z_{1i}-z_{in} \\ x_{1i}-x_{2i} & y_{1i}-y_{2i} & z_{1i}-z_{2i} \\ x_{2i}-x_{3i} & y_{2i}-y_{3i} & z_{2i}-z_{3i} \end{vmatrix} \neq 0 \tag{7}$$

Face i of the polyhedron can be obtained by setting inequality i to an equation.

As an example, orthogonal views of an octahedron are presented in Figure 4. The LP representation of the octahedron would be:

$$\begin{aligned}
-6x \quad\quad +4z &\leq 1 \\
-6y \quad +4z &\leq 7 \\
6x \quad\quad +4z &\leq 55 \\
6y \quad +4z &\leq 49 \\
-6x \quad\quad -4z &\leq -31 \\
-6y \quad -4z &\leq -25 \\
6x \quad\quad -4z &\leq 23 \\
6y \quad -4z &\leq 17
\end{aligned}$$

As would be seen later, it helps to normalize all constraints so that $A_i^2 + B_i^2 + C_i^2 = 1$ for all i. Such normalization of the constraints representing the octahedron results in:

$$
\begin{aligned}
-.832x + .555z &\leq .139 \\
-.832y + .555z &\leq .910 \\
.832x + .55z &\leq 7.145 \\
.832y + .555z &\leq 6.368 \\
-.832x - .555z &\leq -4.029 \\
-.832y - .555z &\leq -3.249 \\
.832x - .555z &\leq 2.989 \\
.832y - .555z &\leq 2.209
\end{aligned}
\qquad (8)
$$

3.0 Machining Pass Generation

3.1.1 LP Representation of Machining Parallel Passes

As was mentioned earlier, machining passes in most situations represent a set of parallel lines scribed on the face to be machined. In a linear algebra context, the passes can be visualized as intersections of a set of parallel planes with the plane of the face. Alternatively, they can also be visualized as the intersection of contours of an objective function with the plane of the face, as the value of the objective function changes from its minimum to its maximum or vice versa (Figure 5).

Since we are interested solely in the lines scribed on the face of interest, we have a wide variety of objective functions which can generate the same set of lines. As seen in Figure 6, planes 1 and 2 intersect with the face plane on the same line. However, computational considerations favor the use of objective functions with certain properties. These are discussed in subsequent paragraphs.

A parameter of interest in the determination of machining passes is the spacing between adjacent passes. We shall now develop a formula for inter-pass spacing and use it in the choice of an objective function.

Suppose the face plane is represented by the equation $Ax + By + Cz = D$ and let the objective function used be $f = ax + by + cz$. Consider two contours of the objective function, f_1 and f_2. The situation is depicted in Figure 7. The distance between the contours, de, is given by [SILVER69]:

$$de = \frac{|f_1 - f_2|}{\sqrt{a^2+b^2+c^2}} \tag{9}$$

The spacing between passes is the distance df which can be expressed as: $df = de \cdot \operatorname{cosec}\theta$, where θ is the angle between the objective function plane and the face plane.

From basic analytic geometry, we have:

$$\cos\theta = \frac{aA+bB+cC}{\sqrt{a^2+b^2+c^2}\ \sqrt{A^2+B^2+C^2}} = \frac{aA+bB+cC}{\sqrt{(a^2+b^2+c^2)}}$$

Hence,

$$\operatorname{cosec}\theta = \frac{1}{\sqrt{1-\cos^2\theta}} = \frac{1}{\sqrt{1 - \frac{(aA+bB+cC)^2}{(a^2+b^2+c^2)}}} \tag{10}$$

Substituting (10) in (9),

$$df = \frac{|z_1 - z_2|}{\sqrt{a^2+b^2+c^2}} \quad \frac{1}{\sqrt{1 - \frac{(aA+bB+cC)^2}{(a^2+b^2+c^2)}}} \tag{11}$$

$$= \frac{|f_1 - f_2|}{\sqrt{(a^2+b^2+c^2) - (aA+bB+cC)^2}}$$

We can have $df = |f_1 - f_2|$ if

$$a^2+b^2+c^2 = 1 \tag{12}$$

and

$$aA+bB+cC = 0 \tag{13}$$

An objective function of the desired form can also be identified directly from the LP tableau of the polyhedron. We shall use an example to illustrate the procedure.

Let us analyze the face of the octahedron corresponding to the second inequality in (8). We set the inequality to an equation and find a basic feasible solution to the linear program by using Phase I of the Simplex method [HADLEY75]. The linear program (sans an objective function) has a slack/surplus variable associated with all but one constraint. Thus, if the polyhedron has m faces, we have in all (m-1) slack/surplus variables and 3 original variables totalling to m+2 variables. The number of constraints equals the number of faces, m. The basic feasible tableau, thus, has m basic variables and two nonbasic variables. In general, the nonbasic variables will be of the slack/surplus type. The basic feasible tableau for the octahedron is shown in Figure 8, where variables s_2 and s_5 are nonbasic variables. If we pick the coefficients of these nonbasic variables in the rows corresponding to x, y and z, we get the negatives of the direction vectors of the two possible movements from the basic solution. In the example of Figure 7, the direction obtained from s_2 is (1/6,0,0) and that obtained from s_5 is (1/12,-1/12,-1/8). Thus, the above directions are two directions in the face plane, and so is any linear combination of the two. If we represent the directions by (d_1,d_2,d_3) and (d_1',d_2',d_3'), then any objective function of the form:

$$f = ax+by+cz \text{ where} \tag{14}$$

$$a = \frac{\lambda d_1+(1-\lambda)d_1'}{\sqrt{\sum_{i=1}^{3} [\lambda d_i+(1-\lambda)d_i']^2}}$$

$$b = \frac{\lambda d_2+(1-\lambda)d_2'}{\sqrt{\sum_{i=1}^{3} [\lambda d_i+(1-\lambda)d_i']^2}}$$

$$c = \frac{\lambda d_3+(1-\lambda)d_3'}{\sqrt{\sum_{i=1}^{3} [\lambda d_i+(1-\lambda)d_i']^2}}$$

satisfies (12) and (13) and spans machining passes in all directions for $0 \leq \lambda \leq 1$.

3.1.2 Location of Parallel Machining Passes

Having chosen a direction for machining passes (i.e., an objective function for the LP), it is necessary to estimate the number of passes required to cover the entire face. At this juncture, it is also necessary to take into account the overlap between adjacent passes, called "cutter overhang". Let us denote the minimum overhang by ε and the diameter of the tool by ϕ.

We, first, need to determine the expanse of the face perpendicular to the direction of the passes. We obtain this quantity, which we may call the "width" (W) of the face by choosing an objective function of the form (14) and minimizing and maximizing it subject to the constraints representing the face. If the minimum and maximum so found are denoted by f_{min} and f_{max}

$$W = f_{max} - f_{min} \tag{15}$$

The width, W, now needs to be divided into a number of strips, representing passes. As shown in Figure 9, the extreme passes have to be located a distance no more than $\frac{\phi}{2} - \varepsilon$ from the extremities of the face. Thus, if n is the number of passes required, the minimum overhang can be ensured by the following inequality:

$$\frac{W-2(\frac{\phi}{2} - \varepsilon)}{n-1} \leq \phi-\varepsilon$$

i.e.,

$$n-1 \geq \frac{W-\phi-2\varepsilon}{\phi-\varepsilon}$$

i.e,

$$n \geq \frac{W-3\varepsilon}{\phi-\varepsilon} \tag{16}$$

Having so determined the number of passes, we segment the face into strips by appending contours of the function (14) in the form of constraints. The contour corresponding to pass j has the form:

$$ax+by+cz = f_j = f_{min} + [\frac{\phi}{2} - \varepsilon + (j-1)\frac{W-\phi-2\varepsilon}{n-1}](f_{max}-f_{min}) \tag{17}$$

The segment of the face between passes j and j+1 is obtained by appending to the original LP the following constraints:

$$f_j \leq ax+by+cz \leq f_{j+1} \tag{18}$$

The tool path can now be determined using the algorithm in Figure 10.

The algorithm generates a sequence of points in the same order in which they need to be traversed. The points can be joined by straight lines to obtain a tool path that spans the entire face.

3.1.3 An Improved Procedure

The algorithm in Figure 10 generates a tool path such as the one shown in Figure 11. As can be easily seen, the tool path does not cover the entire area of the face and some segments near the border of the face remain unmachined. One way of remedying this shortcoming is to allow the tool to move beyond the borders of the face. Such extra movement of the tool can be achieved by applying Algorithm 1 to an enlarged polyhedron wherein all faces except the face being machined are moved outwards (with respect to the polyhedron) by predetermined amounts.

Let us, first, determine the extent to which the tool needs to move beyond the edge in question. As shown in Figure 12, it is adequate to let the tool extend beyond the edge by an amount δ_1 such that the diameter of the tool normal to the pass direction completely clears the face. If ψ is the angle between the edge and the pass direction, the extra movement, δ_1, is given by:

$$\delta_1 = \frac{\phi}{2} \cot\psi \tag{19}$$

The edge can be considered moved "outward" for the purpose of applying Algorithm 1 by an amount δ_2, where

$$\delta_2 = \delta_1 \sin\psi$$

$$= \frac{\phi}{2} \cos\psi \tag{20}$$

Suppose face i intersects with the face being machined at the edge in question. Denoting the inequality corresponding to the face being machined by

$$Ax + By + Cz \leq D$$

the direction of the edge is given by the vector

$$\vec{E} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ A & B & C \\ A_i & B_i & C_i \end{vmatrix}$$

$$= (BC_i - B_iC)\vec{i} + (CA_i - C_iA)\vec{j} + (AB_i - A_iB)\vec{k} \tag{21}$$

The angle between $\vec{E}$ and $\vec{a_i} + \vec{b_j} + \vec{c_k}$ (the vector normal to the pass direction) is the complement of ψ. Thus,

$$\frac{\pi}{2} - \psi = \cos^{-1}\left[\frac{a(BC_i-B_iC)+b(CA_i-C_iA)+c(AB_i-A_iB)}{\sqrt{a^2+b^2+c^2}\ \sqrt{(BC_i-B_iC)^2+(CA_i-C_iA)^2+(AB_i-A_iB)^2}}\right]$$

$$\psi = \sin^{-1}\left[\frac{a(BC_i-B_iC)+b(CA_i-C_iA)+c(AB_i-A_iB)}{\sqrt{(BC_i-B_iC)^2+(CA_i-C_iA)^2+(AB_i-A_iB)^2}}\right] \tag{22}$$

The desired displacement of the intersecting face can now be determined. The relationship between the displacement of the edge, δ_2, and the displacement of the face, δ_3, is brought out in Figure 13 as:

$$\delta_3 = \delta_2 \sin \alpha \tag{23}$$

where α is the angle of intersection between the two faces and is given by

$$\alpha = \cos^{-1}\left[\frac{AA_i+BB_i+CC_i}{\sqrt{(A^2+B^2+C^2)}\ \sqrt{(A_i^2+B_i^2+C_i^2)}}\right]$$

$$= \cos^{-1}(AA_i+BB_i+CC_i) \tag{24}$$

An outward displacement of the intersecting face to an extent δ_3 can be obtained by replacing the right hand side constant, D_i, by the constant, D_i', where

$$D_i' = D_i + \delta_3 \sqrt{A_i^2+B_i^2+C_i^2}$$

$$= D_i + \delta_3 \tag{25}$$

Thus, each constraint i (except the constraint corresponding to the face being machined) needs to be replaced by:

$$A_ix+B_iy+C_iz \leq D_i+\delta_3 \tag{26}$$

where

$$\delta_3 = \frac{\phi}{2}\sqrt{[1-(AA_i+BB_i+CC_i)^2]\{1-\frac{[a(BC_i-B_iC)+b(CA_i-C_iA)+c(AB_i-A_iB)]^2}{(BC_i-B_iC)^2+(CA_i-C_iA)^2+(AB_i-A_iB)^2}\}} \quad (27)$$

3.2.0 Window Frame Milling

In window frame milling, the tool path may be represented by a series of reduced polygons scribed on the surface and may be viewed as moving the tool from one vertex of polygon to another in a fixed direction (Figure 14). This may be visualized as moving from one basic feasible solution, which corresponds to an extreme point of the feasible region, to an adjacent one, when the face is represented by LP representation. When a face of a polyhedron is selected (the corresponding inequality is set to equality), the basic feasible solution would always have two non-basic variables. If one of the two non-basic variables are selected as the entering variable, a simplex pivot would move the current solution from one vertex to an adjacent solution on the face. A judicious choice of the entering variable at each pivot can move the solution completely around the periphery of the polygon.

When the tool completes machining along the outermost polygon, it has to move onto the next inner polygon. This can be done by moving the planes an equal distance inward except the face to be milled. This is done by subtracting some amount δ_3 from the right hand side, or

$$A_i x + B_i y + C_i z <= D_i - \delta_3 \quad (28)$$

$$\delta_3 = d.\sin(\alpha)$$

$$= d\ \{1 - (A\ A_i + B\ B_i + C\ C_i)^2\}^{1/2} \quad (29)$$

where d: distance to be moved.

d = 0.1 (tool diameter) for outermost polygon, and

d = 0.6 (tool diameter) for the rest of reduced polygons.

As the polygon is being reduced repetitively, the feasible region of the milling face is reduced gradually, and finally disappears. When this happens, the LP problem becomes infeasible, indicating that the entire face has been covered or the uncovered face area is too small for a complete pass but a last pass is needed. The non-existence of a feasible region may be detected in Phase I of Simplex Algorithm when a positive artificial variable exists in the basis.

The above procedure is illustrated in Figure 14, and may be summarized in the following:

1. Move planes equally inward by (D/2 - ε) to insure the minimum overhang.

2. Find the basic feasible solution by using Phase I of the Two Phase Simplex Algorithm. If a positive artificial variable exists in the basis, the feasible region does not exist. Go to step 6. Otherwise, a feasible starting basis is found. Record the extreme point and go to next step.

3. Apply Phase II of Two Phase Algorithm to the new tableau. Pick the entering variable from the two non-basic variables (the variable which just left the basis cannot be entering variable in the next pivoting).

4. Record the new extreme point. If this point is not the starting point go back to step 3. Otherwise, continue to next step.

5. Move the planes inward by the amount of (D - ε) except the face to be milled using Equations (28) and (29). Go back to step 2.

6. Move the planes outward by the amount of (D/2 - ε) except the face to be milled using Equations (28) and (29).

7. Apply Phase I as in step 2. If a positive artificial variable exists in the basis, problem is infeasible. Go to step 9.

8. Repeat step 3 and 4.

9. Machining path generation is completed, stop.

4.0 Conclusions

This paper presented a procedure to represent a polyhedron by a linear program to facilitate the machining of its faces. LP representation helps eliminate in the redundancy of information as is typical with an IGES-type model and paves the way for research in the area of tool path optimization.

REFERENCES

HADLEY75: Hadley, G., LINEAR PROGRAMMING, Addisoin Wesley Publishing Company, Reading, 1975.

SILVER69: Silverman, R. A., MODERN CALCULUS AND ANALYTIC GEOMETRY, The Macmillan Company, New York, 1969.

TAHA76: Taha, H. A., OPERATIONS RESEARCH - AN INTRODUCTION, Macmillan Publishing Company, Inc., New York, 1976.

TEICH85: Teicholz, Eric, (Ed.), CAD/CAM HANDBOOK, McGraw-Hill Book Company, New York, 1985.

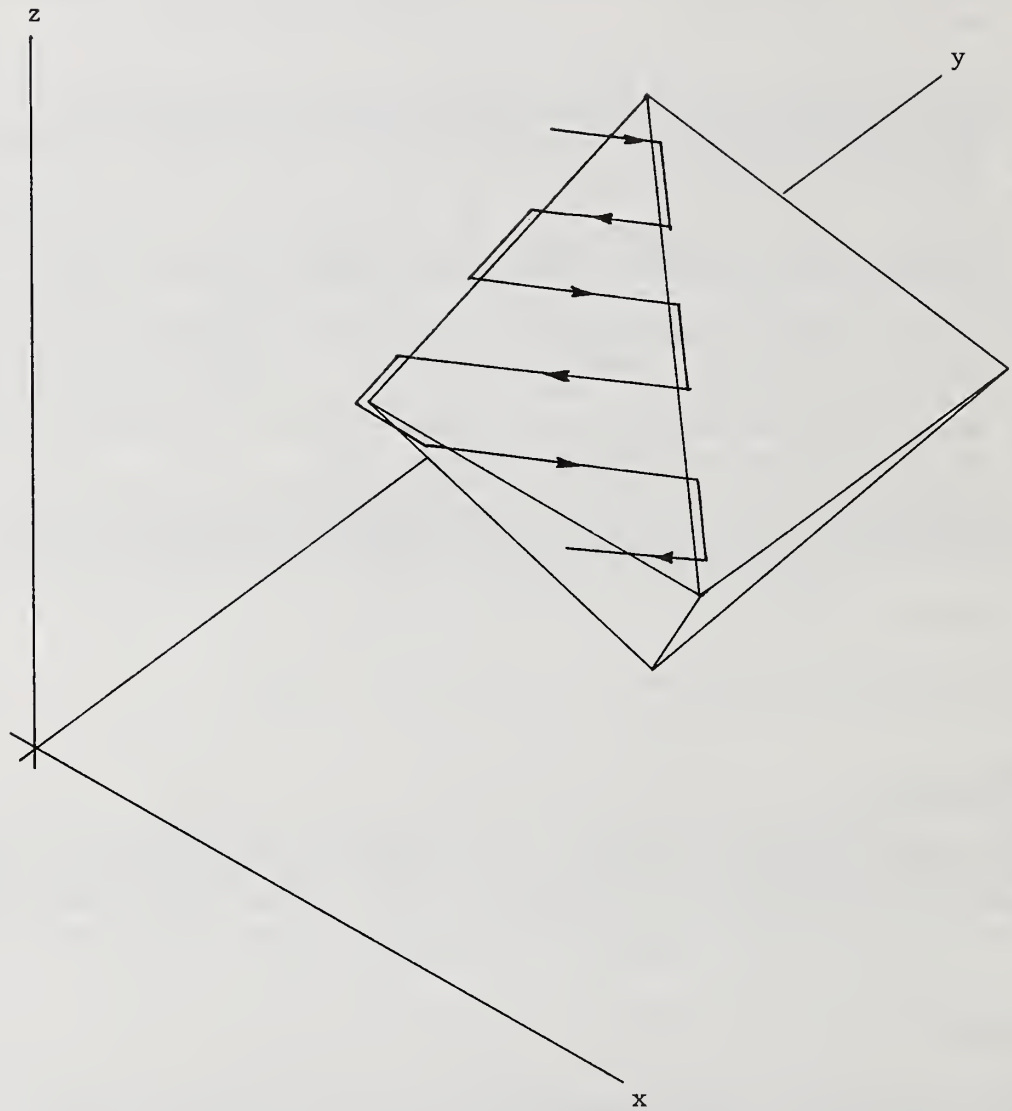


Figure 1: Machining Passes on a Face

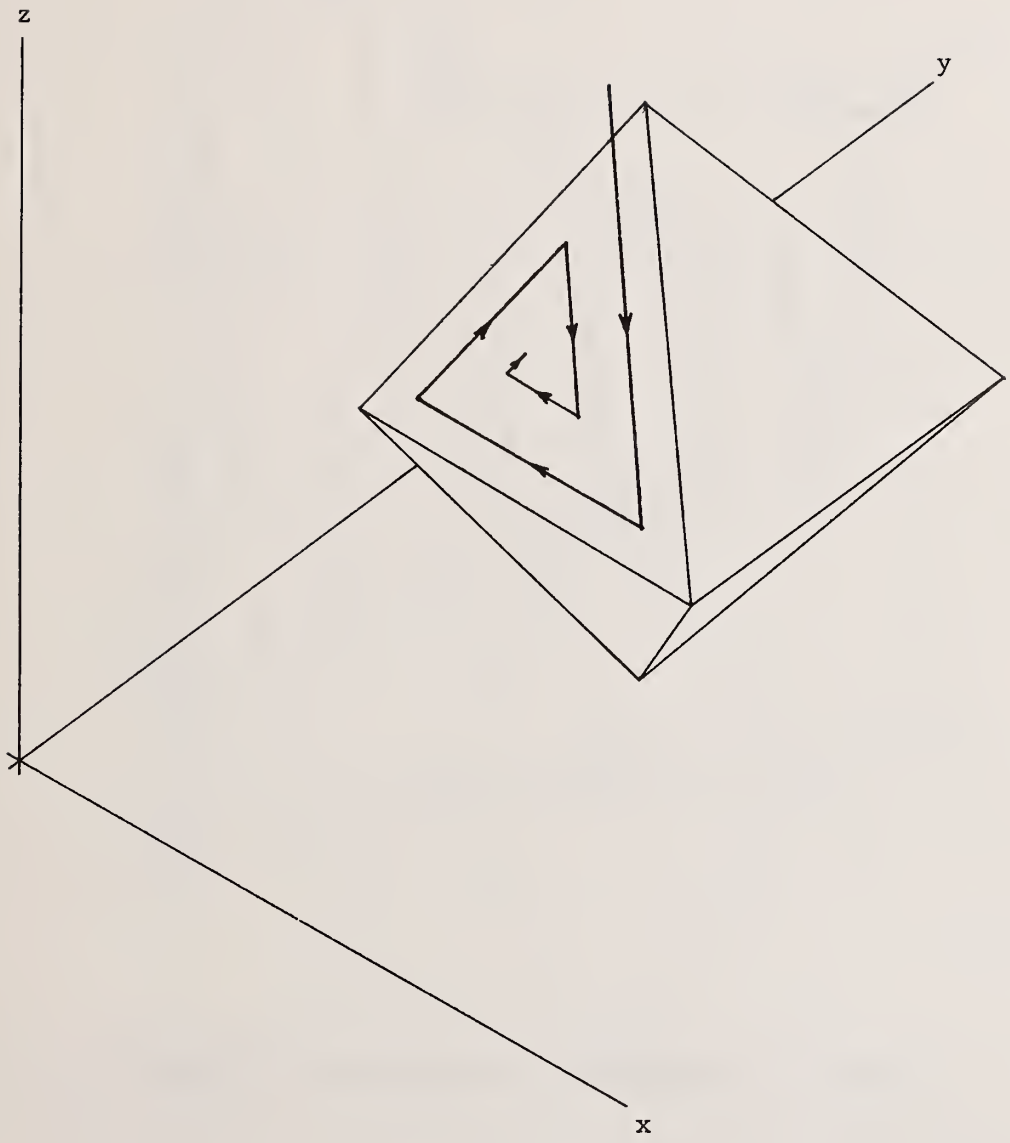


Figure 2: The "Window-Cutting" Technique

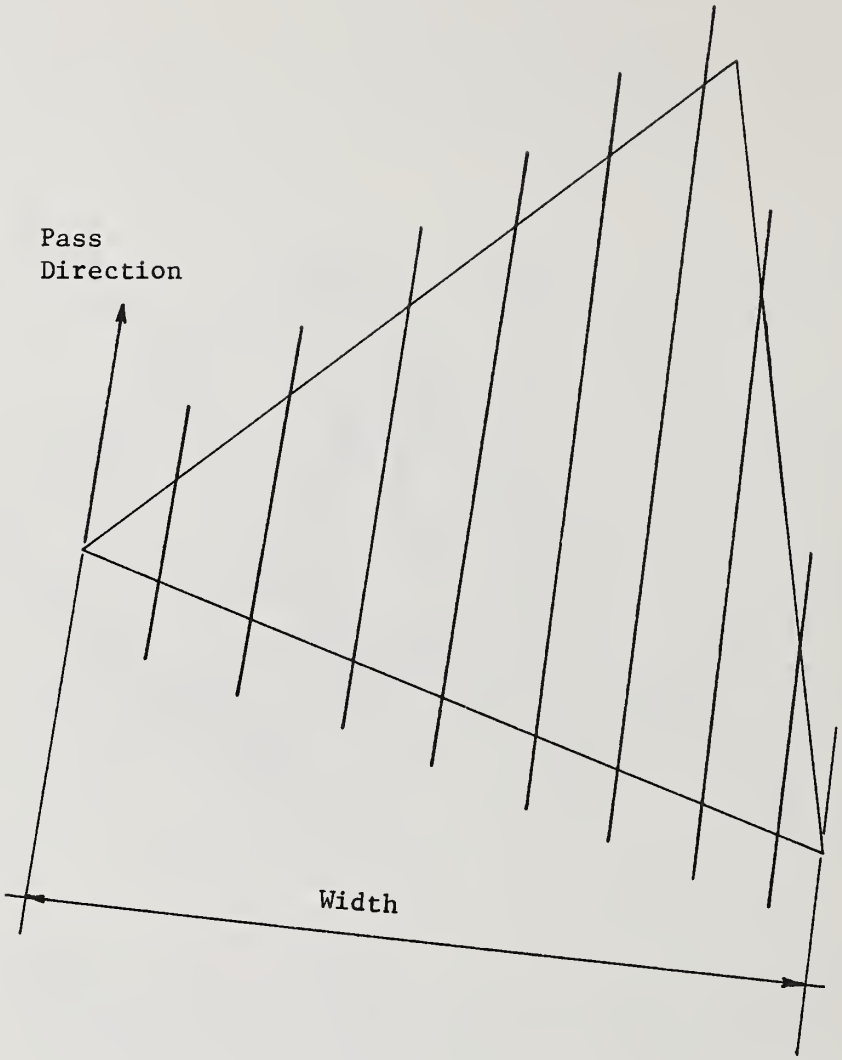
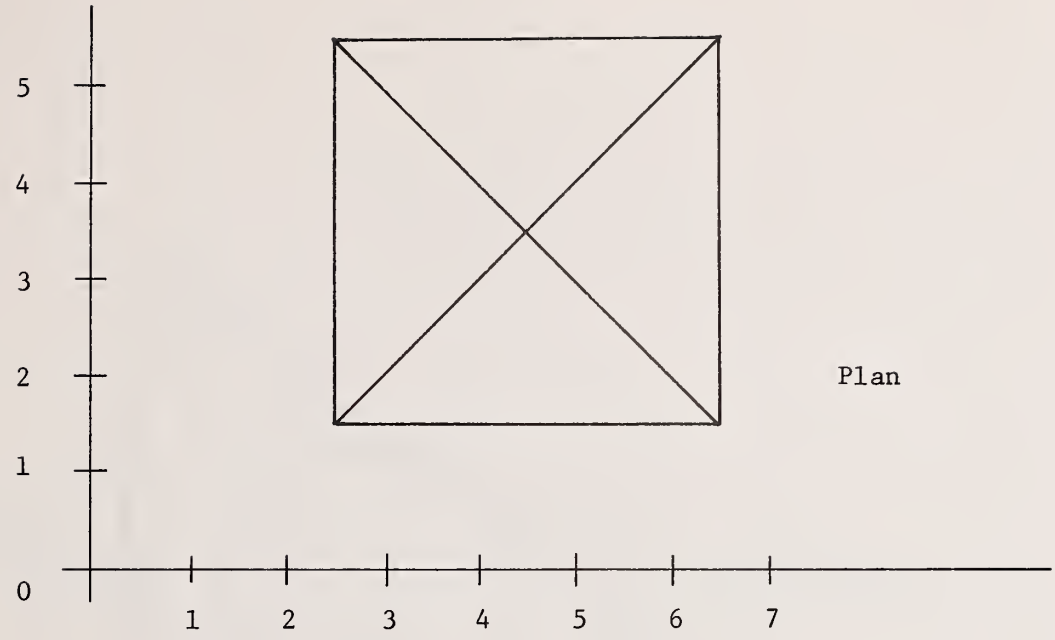


Figure 3: Determination of the Number of Passes



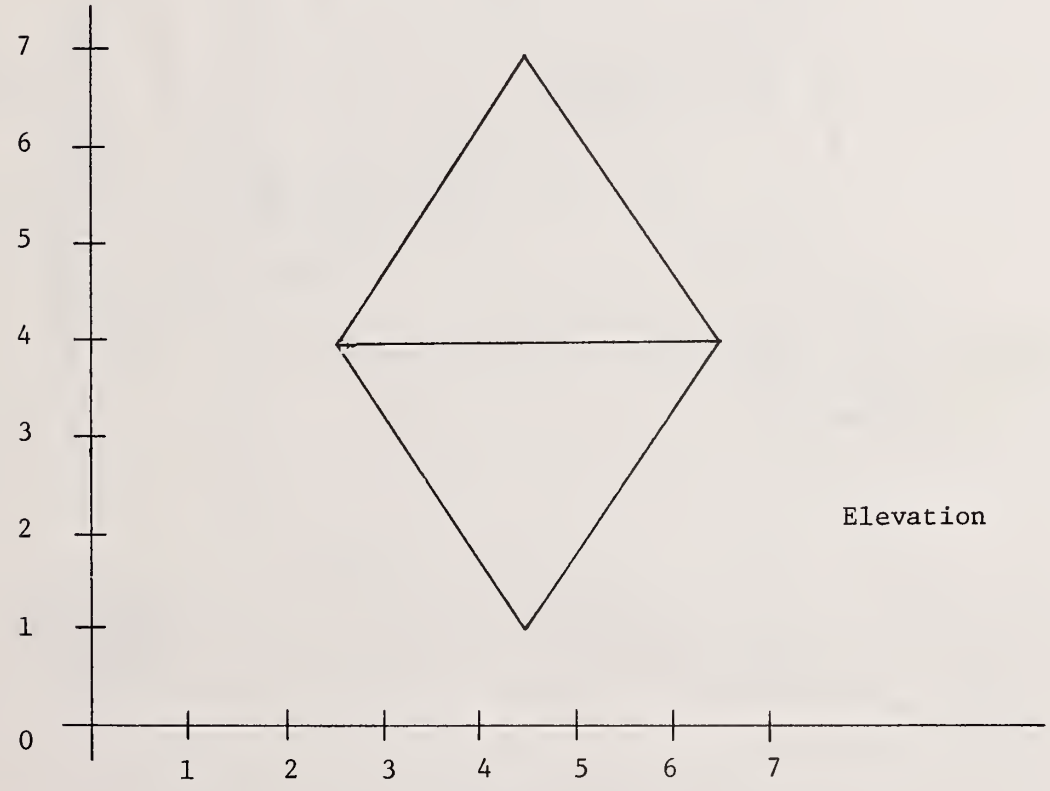


Figure 4: Orthogonal Views of an Octahedron

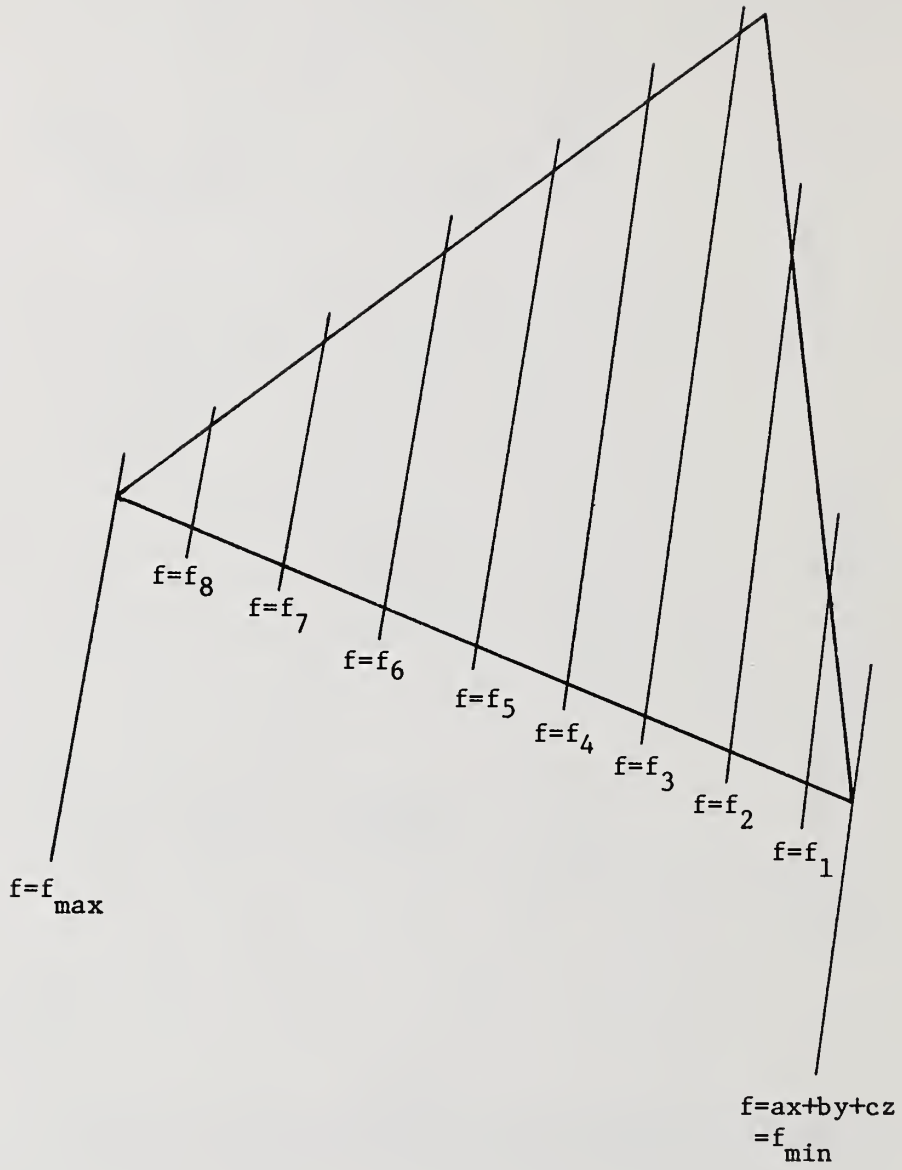


Figure 5: Objective Function Contours on a Face

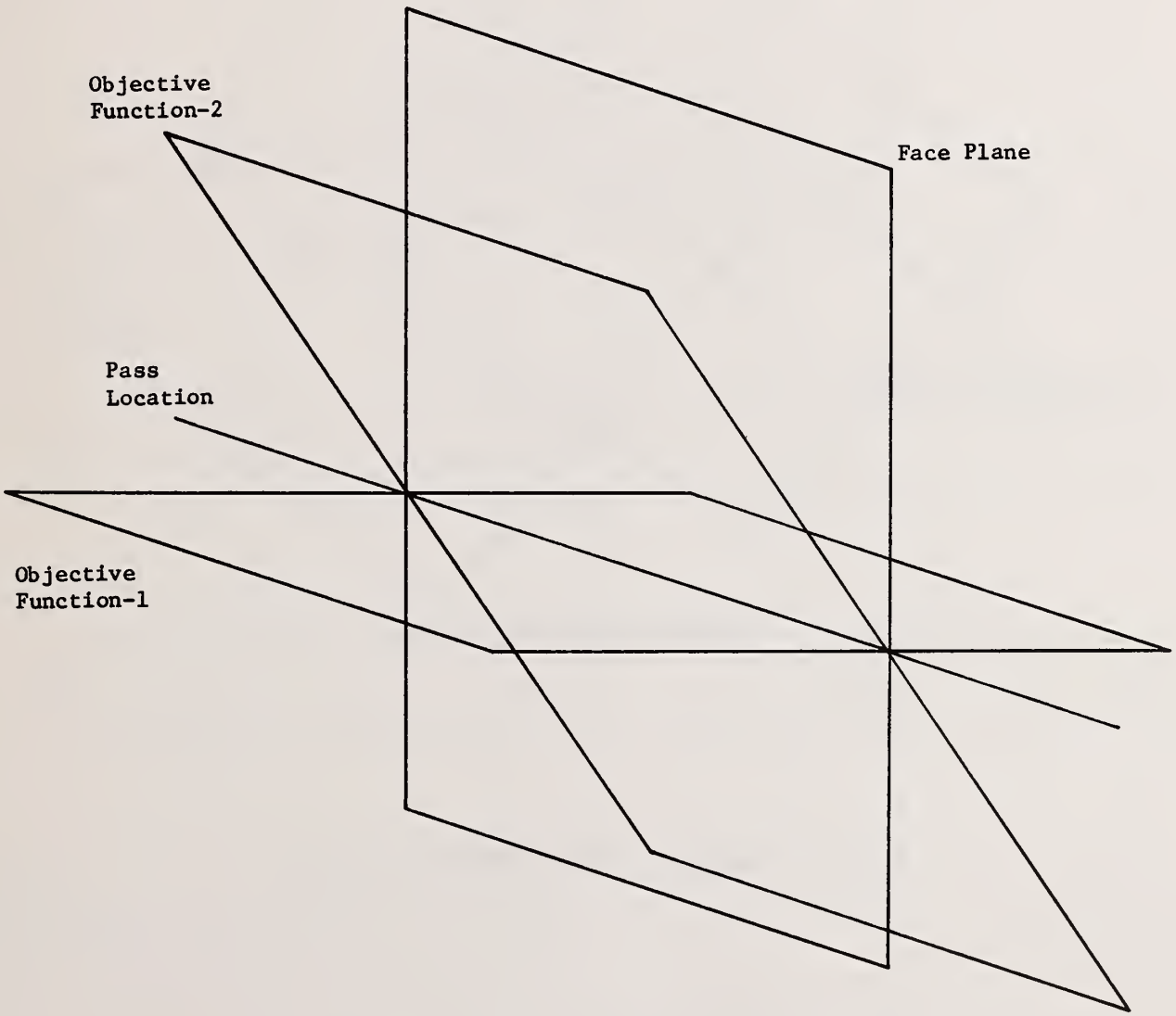


Figure 6: Multiplicity in the Choice of an Objective Function

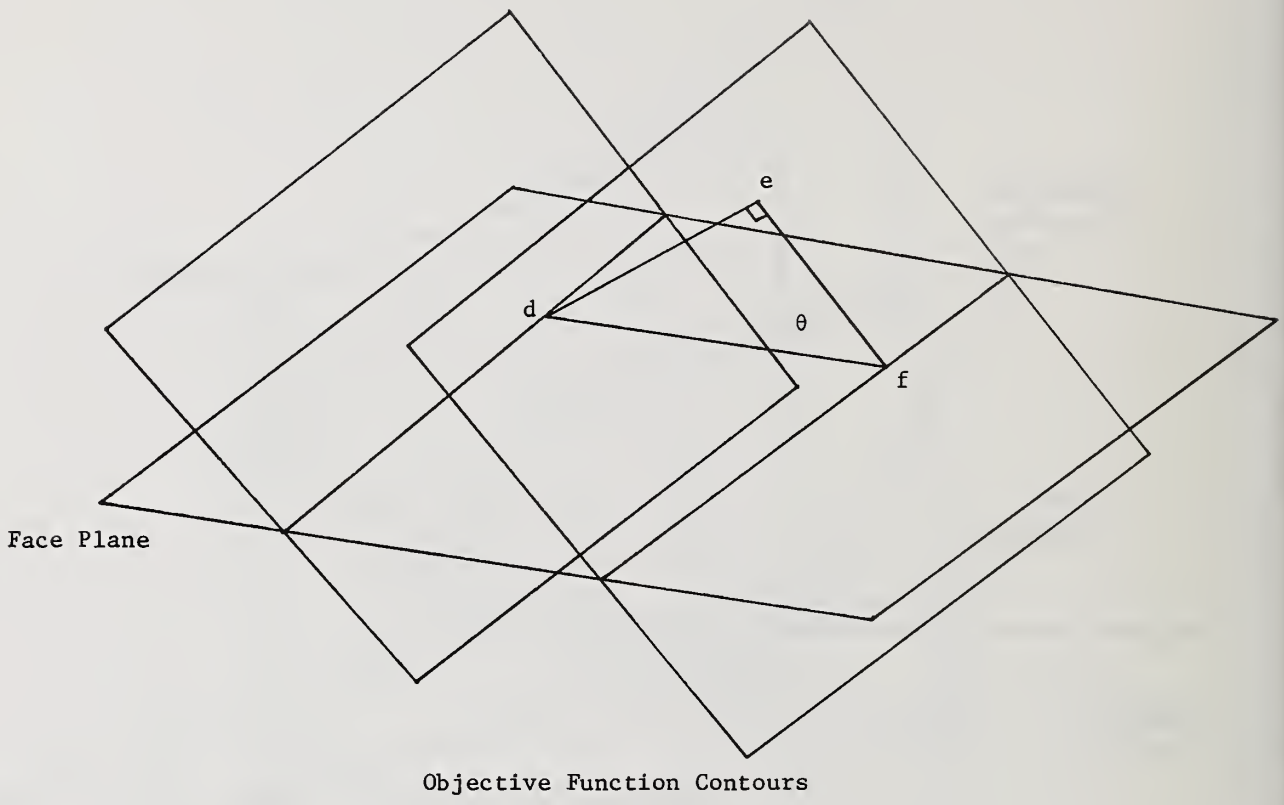


Figure 7: Determination of Inter-Pass Spacing

Basic Var.	x	y	z	s_1	s_2	s_3	s_4	s_5	s_6	s_7	13/2
s_1				1	1			1			24
s_4					1		1				24
y		1						-1/12			3/2
s_3						1		1			24
z			1					-1/8			4
S_6					-1			-1	1		0
x	1				1/6			1/12		1	13/2
s_7											24

Figure 8: A Basic Feasible Tableau

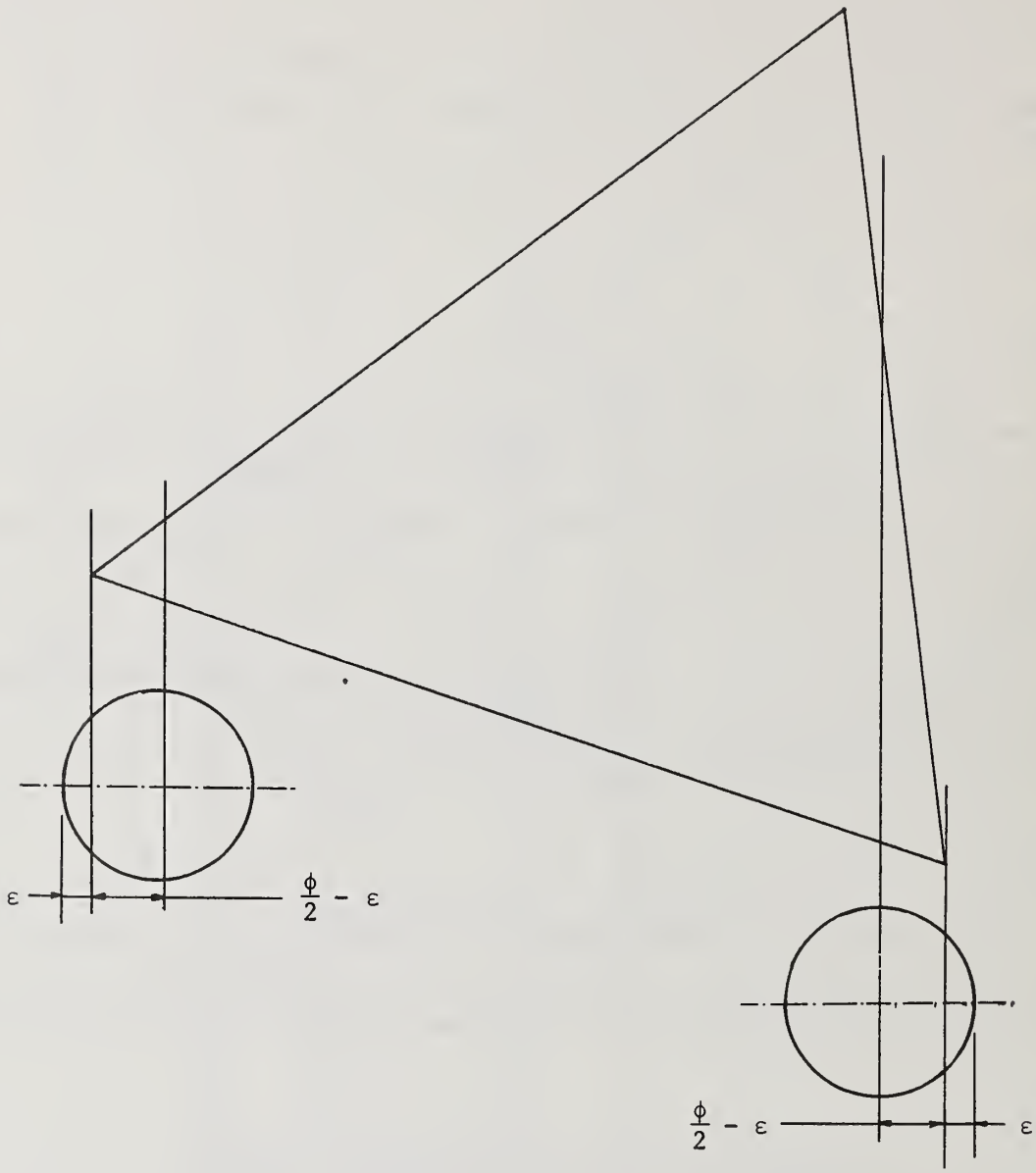


Figure 9: Location of Extreme Passes

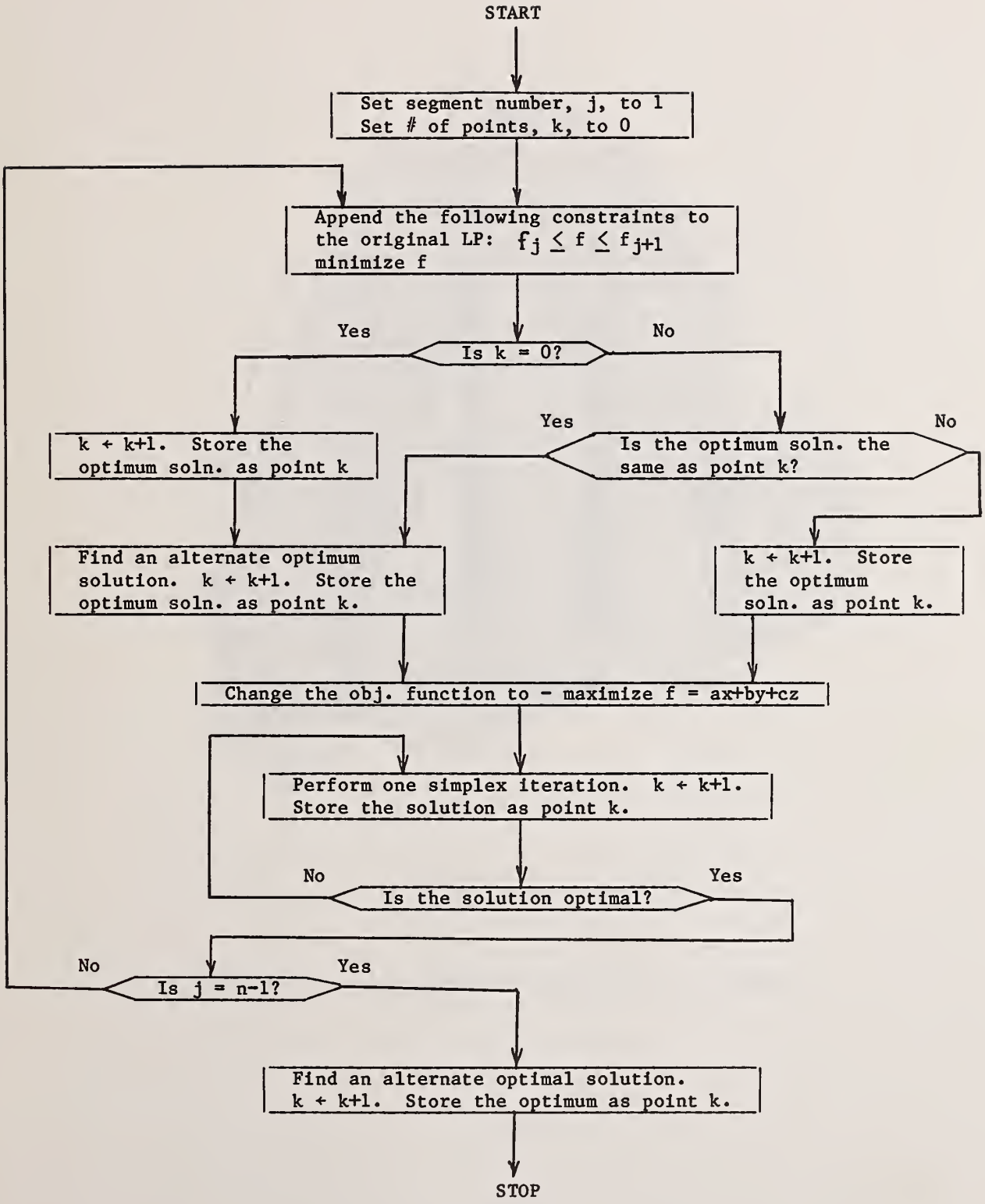


Figure 10: Algorithm to Generate Tool Path

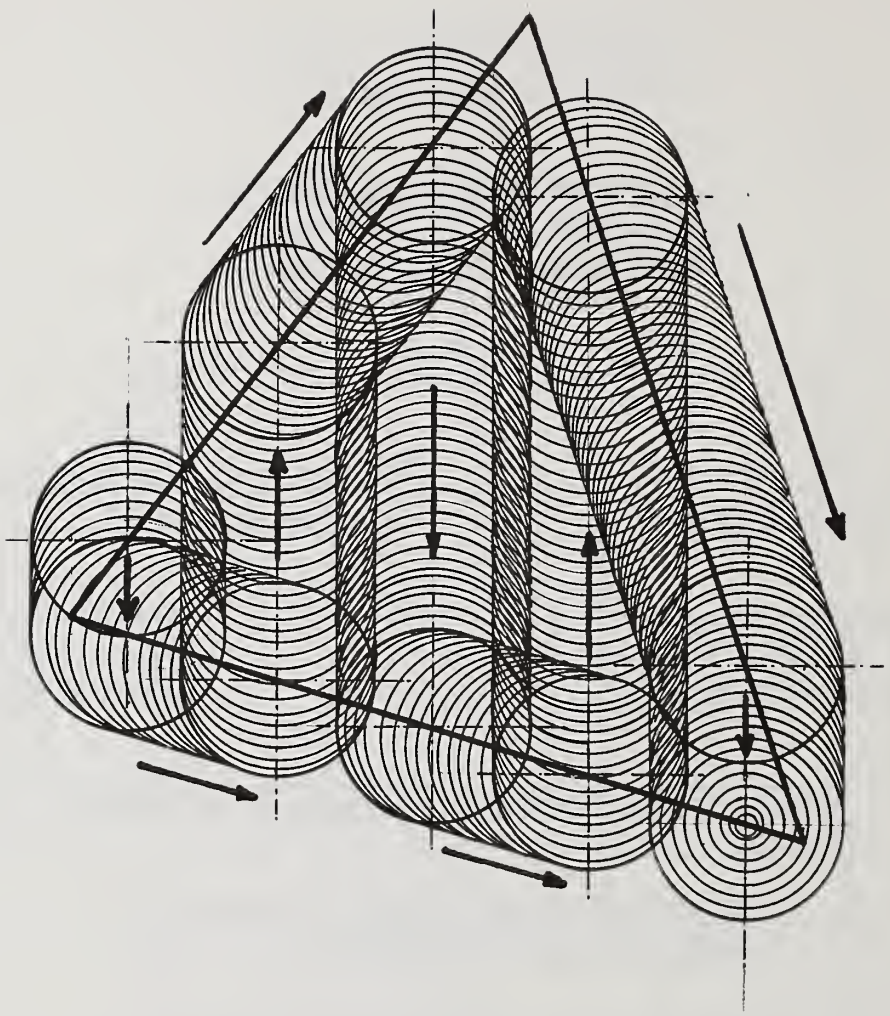


Figure 11: Tool Path Generated by the Algorithm

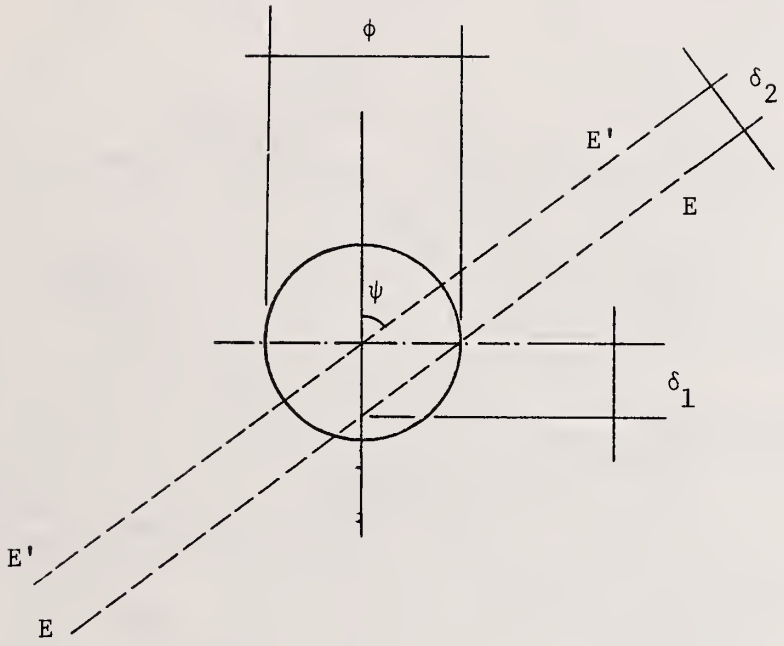


Figure 12: Displacement of Face Boundaries

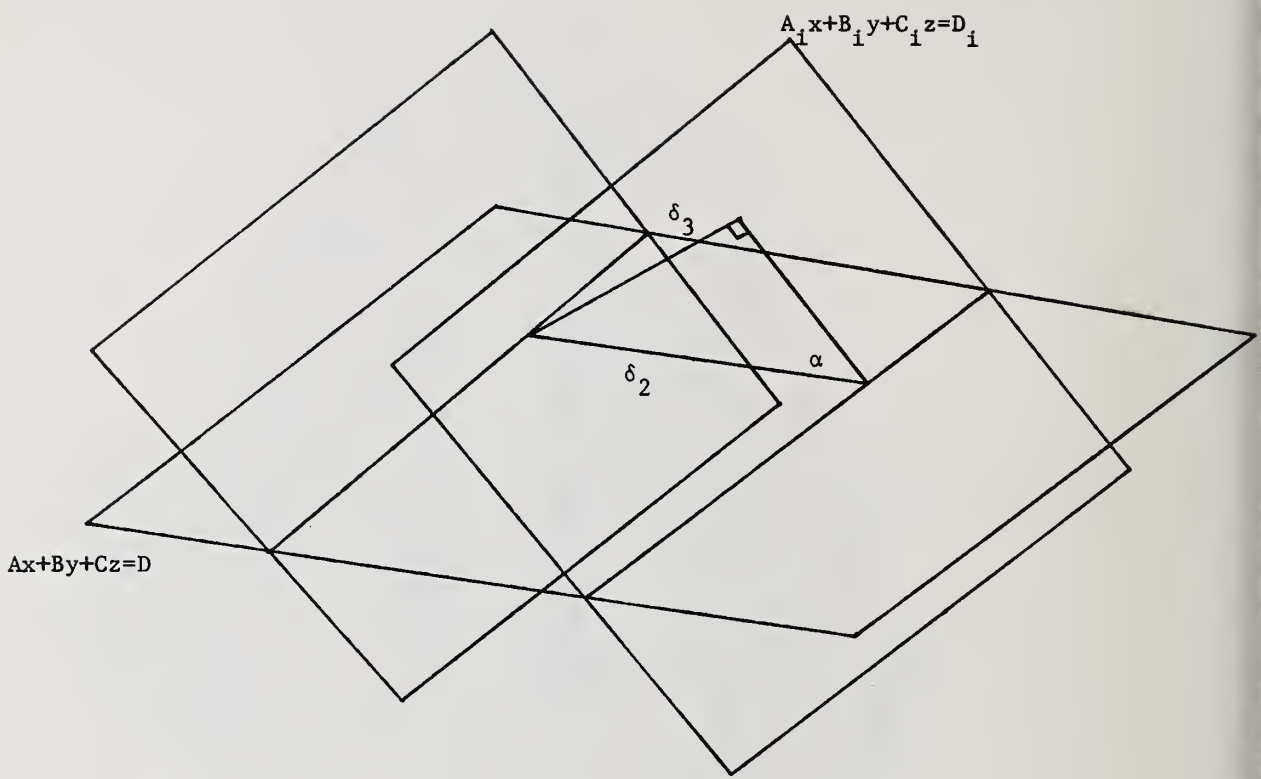


Figure 13: Displacement of LP Constraints

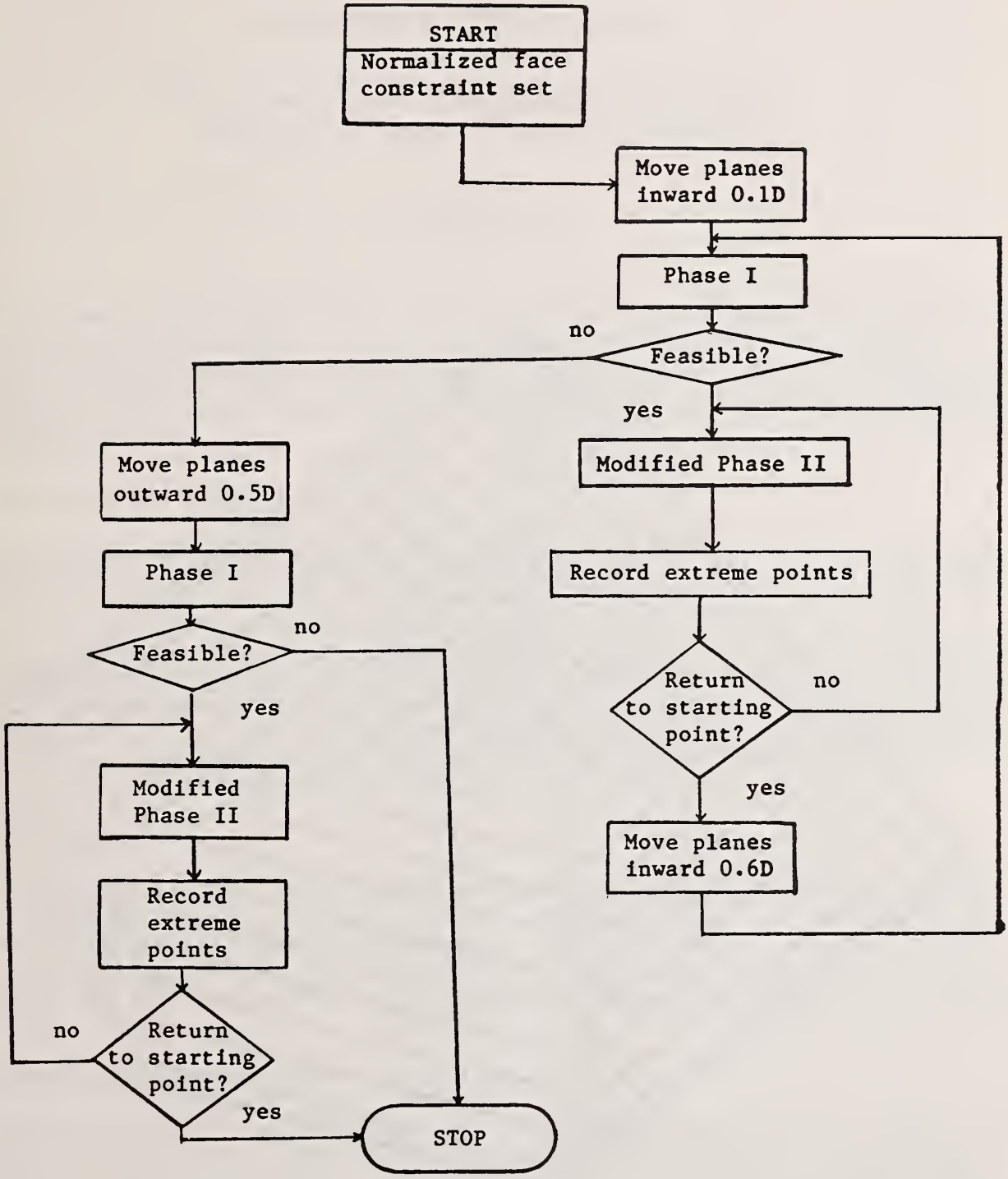


Figure 14 Algorithm to Generate Window Cutting Tool Path
(Minimum tool overhang is 0.4 tool diameter)

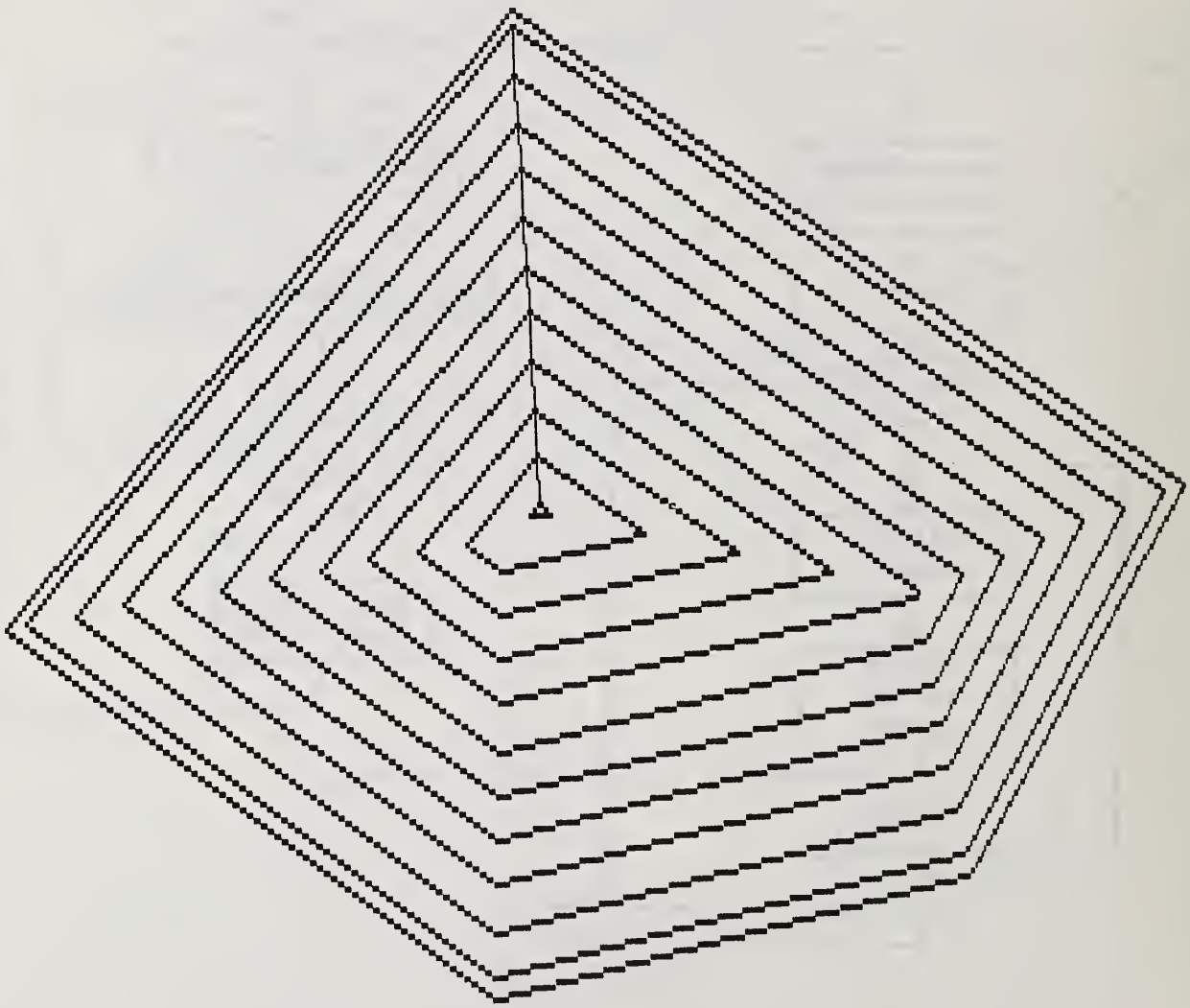


Figure 15: Tool Path Generated by the Window Cutting Algorithm

SYSTEM BUFFERS REQUIRED WHEN FMS ARE USED IN FABRICATION & ASSEMBLY OPERATIONS

DR. William P. Darrow, Department Of Management,
Towson State University, Towson, MD 21204

INTRODUCTION

The purpose of this paper is to review sources of uncertainty and strategies for using system buffers to stabilize the materials management system in a fabrication and assembly shop using conventional machining technology. The buffers needed to take advantage of FMS technology will then be examined. Finally a conceptual framework for integrating the two technologies will be presented.

Fabrication and Assembly Operations

Fabrication operations refer to the production of discrete component (single piece) parts from raw materials. Assembly operations consist of both the subassembly and the final assembly operations needed to produce the end product. A large number of industrial organizations include both types of operations in the same plant. These companies produce a wide range of products for industrial, consumer, and defense applications. This industrial sector is of interest because it was the first to embrace Material Requirements Planning (MRP), and is currently introducing Flexible Manufacturing Systems (FMS) technology. Therefore, fabrication and assembly operations will provide the setting for the observations and analysis that follows.

Terminology

The term "MRP" will be used to represent both the original concept of Material Requirements Planning systems, and the expanded concept of Manufacturing Resource Planning systems (sometimes referred to as MRP II). The broader concept includes all of the subsystems of the overall materials management system. The meaning of the term in any given instance will be clear from the context.

Buffering in Materials Management Systems

The term "buffer" was first used by materials managers in the context of safety stock. Safety stock provides a buffer, or a means of absorbing the shock of unforeseen customer demand. This buffer protects the system against the uncertainty of customer demand by providing goods to be used to satisfy unexpected demand, without creating a stock out. A stock out can be thought of as an instability in the inventory system. Stock outs trigger an expedited replenishment order, which must be processed more quickly than would be allowed by the normal lead time. Instabilities such as this are undesirable, as they consume managememt's attention, and often result in added costs.

UNCERTAINTY IN MRP SYSTEMS

The problem of uncertainty in MRP systems will be discussed by identifying the many sources of uncertainty. A useful classification scheme will then be presented to gain further insight, along with some related research results.

Sources of Uncertainty in MRP Systems

The concept of buffering mechanisms was greatly extended in response to problems encountered by the users of Material Requirements Planning systems. Eichert (4) identified a number of sources of uncertainty affecting MRP systems. He suggested that many of these sources of uncertainty could be handled by the system. Eichert recommended combining a number of the sources of uncertainty as a single independent demand source, and then including the resulting demand in the Master Production Schedule. Steele (11), and later Mather (9), related sources of uncertainty to the problem of MRP system sensitivity. They suggested strategies to minimize the disruptive effects of uncertainty, which will be discussed below. The sources of uncertainty identified by these three writers is summarized below in Table 1.

TABLE 1. Sources Of Uncertainty In MRP Systems

Source of Uncertainty	Eichert	Steele	Mather
MPS Changes	X	X	X
Service Parts	X	X	
Field Failures	X		
Vendor Shortages	X		X
QC Rejects	X		X
Process Scrap	X		X
Pull Ins	X	X	
Eng. Changes	X		X
R & D Demand	X		
Record Keeping Errors	X	X	X
Parameter Changes		X	X
Mach. Breakdowns			X

Categories of Uncertainty in MRP Systems

Whybark and Williams (15) provide an interesting perspective on uncertainties in materials management systems. They recognized two basic types of uncertainty, one for timing and one for quantity. They further recognized that each type of uncertainty could be further classified as to whether the source of the uncertainty related to supply or demand. Taken together, this defines four categories of uncertainties in MRP systems. These categories are shown below in Table 2, with illustrative examples for each.

Whybark and Williams developed a simulation model to evaluate the use of safety stock and safety lead time in MRP systems. They concluded that the use of safety lead time is most appropriate for situations involving uncertainty in timing, and that safety stock is the most appropriate buffer for system uncertainties relating to quantity. This research raises some interesting

questions regarding safety stock for order point inventory systems. Independent demand items are readily modeled in MRP systems by using the Time Phased Order Point technique. Following this line of reason, the classical order point inventory system can be thought of as a special case of MRP. This suggests that the results of Whybark and Williams study might apply to order point systems.

TABLE 2. Categories Of Uncertainty In MRP Systems

\SOURCE TYPE\	SUPPLY	DEMAND
TIMING	Changes in Lead Time by the Vendor or Shop	Customer Changing Due Dates, Changes in MPS, Early Order Release to Level Work Load
QUANTITY	Process Yield, Quality Control Rejects, Eng. Changes, Inventory Record Accuracy	Errors in Customer Forecast, Changes in Production Plan or MPS, Changes in Parameters

NERVOUSNESS IN MRP SYSTEMS

"Nervousness" in MRP systems occurs when the MRP system logic amplifies an otherwise minor change in the system. The result is an inordinately large volume of action and/or exception messages. This in turn places a great burden on the schedulers, planners, and operating personnel who are needed to make the system work. The result is a decrease in shop performance. If left unchecked, an unacceptable level of systems sensitivity can eventually lead to the abandonment of the formal system itself. Adequate system buffers can reduce this problem to a manageable level. "Nervousness" in MRP systems was first defined by Steele (11), who identified a number of causal factors, and recommended several strategies to overcome the problem. Later, Mather (9) added several other strategies to minimize instabilities in MRP systems.

The materials planner can use a number of strategies to insulate himself from sources of instability. Steele recommends: 1) minimizing changes to the MPS, 2) controlling parameter changes, 3) using Pegging and the Firm Planned Order to circumvent lot sizing logic, and 4) relating the lot sizing strategy to the product structure (to be discussed below). Mather adds to this list of strategies by recommending : 1) minimizing record keeping errors, 2) avoiding the use of dynamic lot sizing rules, 3) freezing the near term portion of the MPS with a Time Fence, 4) using safety stock, 5) forecasting requirements for service parts, 6) using quality assurance techniques to minimize yield and quality losses, 7) stressing delivery performance in vendor evaluation, 8) treating vendors as external work centers 9) using preventative maintenance, and 10) using a Block Change policy to introduce engineering changes. These strategies will be discussed in the sections below, along with more recent contributions to the problem's resolution.

Stabilizing the MPS

The Master Production Schedule provides the framework for all manufacturing operations in fabrication and assembly plants. Stabilizing the MPS can exert a great influence on stabilizing the entire system.

The Master Production Schedule drives the MRP system. Changes in the Master Production Schedule can occur either in response to changes in the overall production plan, or as the MPS is fine tuned to reflect adjustments in the timing and quantities for specific end items. The design of the larger system is such that any change in the MPS will be passed down to the MRP system, then to Capacity Requirements Planning, and ultimately to the shop floor's Production Activity Control system. Therefore, stopping an unnecessary change at the level of the MPS eliminates many "downstream" changes in subsequent systems.

Time Fences are one technique for managing changes to the MPS. The Time Fence "freezes" changes for the periods within the fence, at the near end of the planning horizon. This is actually a policy that limits the system in making automatic changes to the MPS in the near term, where they are most likely to disrupt the execution systems. Depending on the system, automatic changes may be restricted in terms of both quantity and timing, or be limited to changes in timing only. It is also possible to use two Time Fences, one fence for fixed quantities, and another for restrictions on both timing and quantity. Inside the Time Fence(s), the restricted changes can be overridden manually, with the mutual consent of both operations and materials management.

Lot Sizing

Lot sizing policies are responsible for creating many of the sensitivity problems in MRP systems. Lot sizing can cause changes in quantity and timing to pass from level to level in the product structure. Dynamic lot size policies can create instabilities on any given level in the product structure. In this section strategies for minimizing these problems will be discussed.

Lot sizing is one of the main causes of nervousness in MRP systems. This is because a number of different sources of uncertainty can trigger a lot sizing recalculations. When lot sizing policies are applied to assemblies or subassemblies, a change at any level, higher than the component level, can cascade down to lower levels. For example, if quality control rejects a subassembly, the resulting lot sizing logic may generate a gross requirement far in excess of what is actually needed. MRP logic then proceeds to compound the error by generating requirements for all of the related lower level items. This process continues to cascade downward until the lowest levels in the product's structure are reached.

Dynamic lot sizing policies such as Least Total Cost, Part Period Balancing, or Period Order Quantity can also cause system instabilities by triggering a series of lot sizing recalculations along the planning horizon. Recalculations can be triggered by any, otherwise minor, change the system that effects either the quantity or the timing of the net requirements. Something as simple as a minor adjustment to an inventory record, a service part requirement, or a change in planned lead time could trigger a chain reaction with dynamic lot sizing. This source of instability is different from the cascading effect described above, and can occur at the component level. If dynamic lot sizing

is used at other than the lowest (component) levels in the product structure, chaos can result, as the effects of dynamic rescheduling and the cascading effect are combined. Dynamic lot sizing logic can produce unanticipated results. Steele (11) and Mather (9) have each given an example of a decrease in the Master Production Schedule leading to a severe increase in requirements at lower levels.

The Firm Planned Order and Pegging can be used to override lot sizing logic. Pegging is the capability to trace parent/child logical relations between material requirements at various levels in the product structure. The Firm Planned Order is a technique that overrides the computers automatic rescheduling of Planned Orders. The timing and the quantity of a Firm Planned Order is fixed by the planner, and will not be changed by the computer. The combination of these two techniques can defeat automatic rescheduling, and can prevent the problem of "cascading", where lot sizing multiplies requirements as they are passed down from level to level in the product structure.

Tailoring lot sizing strategy to the level in the product structure can help reduce nervousness in the system. Theisen (12) first recognized the relationship relating lot size to product structure. He recommended fixed quantity policies be used for the MPS level, and Lot-For-Lot policies for subassemblies at the intermediate levels in the product structure. For component parts, which do not exhibit the cascading effect and have relatively high setup costs, Theisen recommends use Least Total Cost. Steele (11) suggests a similar strategy where fixed quantity policies are used for the top level items, fixed quantity or Lot-For-Lot policies for intermediate level items, and Period Order Quantity for component parts.

Parameter changes and record keeping errors can interact with lot sizing logic to trigger a chain reaction of system action messages. Changing the lead time or the level of safety sock required can trigger a chain reaction of requirements. Record keeping errors in stock levels, or in the bill of material structure can lead to unexpected shortages.

Current research in lot sizing and system sensitivity has attempted to deal explicitly with the cost of MRP system nervousness. Kropp, Carlson, and Jucker address system instabilities by developing a modification of the Wagner-Whitin (13) dynamic lot sizing model to include the costs of changing setups in response to system instability (2, and 7). Their original model has been refined by Kropp and Carlson (8) to include a penalty cost for cancelling a previously scheduled setup. As pointed out by Collier (3), there are two obvious problems with this approach. First, as with many of the parameters in mathematical inventory models, it is very difficult for accurate cost estimates to be developed. Consider the problems faced by a cost accountant or an industrial engineer asked to determine the cost of cancelling a previously scheduled setup. Second, the extended Wagner-Whitin model has an even greater computational burden than the original. Since the original model has not met with acceptance by the practitioner, according to a recent survey of MRP system design by Wemmerlov (14), it is unlikely that the more complex extensions will meet with acceptance by practitioners.

Other Issues in MRP System Sensitivity

There are a number of sensitivity issues that are unrelated to lot sizing. In

this section they are identified, and system design approaches for minimizing their impact are given.

Engineering Change Notices can lead to numerous action messages and create problems with system stability. Mather (9) suggests that fabrication and assembly operations adopt the "Block Change" policy used by the automobile industry. Under this policy, engineering changes are batched (or "blocked") and released to operations at regular predetermined intervals.

Service parts and field failures can create unexpected demand in MRP systems. Service parts are best treated as independent demand items, using the Time Phased Order Point technique. Field failures are the infrequent breakdowns of key production equipment that have severe economic consequences for the customer. The best way to anticipate this is by maintaining a spare parts inventory of critical service parts.

Vendor shortages can be controlled in a number of ways. Vendor delivery performance should be a primary consideration in vendor evaluation. It should be ranked right along side of price and quality in the vendor selection process. Quality problems can be detected as soon as they occur by working with the vendor to develop source inspection procedures. Critical vendors should be provided with as much forward visibility as possible by being apprised of the production plan well in advance of the actual purchase order release. Blanket purchase orders can also be used to "reserve" capacity in the vendor's plant.

Machine breakdowns can stir up a lot of action, both on the plant floor and in the planning department. Strategies to minimize this source of uncertainty include preventive maintenance and maintaining an inventory of critical (long lead time) spare parts. No matter how well you plan to prevent breakdowns, and maintain critical spares, there will be machine outages. When the unexpected happens, and it always does happen in manufacturing, it is useful to have up-to-date information on vendors who can provide a backup for the plant's manufacturing processes.

Lead times directly effect system sensitivity in several ways. The lead time for the end item is the sum of the lead times along a critical path in the bill of materials structure. A change in the lead time of any item along the critical path can change the lead times for the end item, all of its subassemblies, and all of its component parts. Thus any changes in lead time can trigger a chain reaction of rescheduling. Lead times explicitly, or implicitly, include an allowance for queue times at each work center. Lead times can be represented by a linear function of the queue allowance multiplied by the number of operations, plus setup and run times. Since the sum of the setup and run times are constant for a given lot size, lead times are directly related to the allowance for work center queues. These queues of work in process serve as the principal buffer the Production Activity Control system.

Another link to system sensitivity is provided by Kanet (6), who points out that keeping lead times short reduces the item's exposure to potential changes in timing and quantity, and the subsequent need to reschedule.

Safety lead time gives the planner an alternative to safety stock. This may be a useful trade-off for purchased items, as Whybark and Williams have suggested

that safety lead time may be more effective than safety stock when faced with uncertainty in timing (15). However, because of the relationship between lead time and queue time, safety lead time will lead to inflated levels of work in process when applied to shop orders. Safety lead time also undermines the shop's priority control system.

Yield allowances are necessary to guard against variations in process losses. Their purpose is to assure that an adequate supply of component parts will be available to produce the finished goods required by the MPS. Yield allowances should not be designed to provide protection against serious quality problems, which may lead to the rejection of the entire lot. Safety stock is the appropriate technique to provide protection against the latter problem.

Safety stock is a common buffering technique used in independent demand systems. Orlicky (10) does not recommend safety stock for dependent demand items. Under a policy of zero safety stock for components and subassemblies, some inventory will be available as a result of variations in quality and process yield. When quantities are greater than expected, the excess will be placed in stock. When quantities are less than expected, the result will be a shortage in final assembly. The work already in process for the order that is shorted will be completed and placed in stock. However, the stock generated as a result of yield allowances will not provide much protection against a serious quality problem. Under a zero safety stock policy, any excess stock will be applied to the next order, and will not be allowed to accumulate. In any event, the lack of planned safety stock at component and intermediate levels will not affect customer service levels if a safety stock of finished goods is maintained.

Queues in MRP systems are a natural product of the functional shop layout, where machines are grouped together by function. Queues of work are necessary to assure adequate machine and labor utilization. This layout is markedly different from a flow line, or process layout, where the machines are laid out to follow the order of the product's process steps. In a process layout, machine output is balanced. As a result, minimal queues are needed to maintain high labor and machine utilization. In a functional layout variations in processing create a haphazard, nearly random, flow of material through the shop. Processing times are not synchronized between work centers. The scheduling and coordination problems would be impossible to deal with if queues were not used. With queues of work at each work center providing a buffer, the work centers can be treated independently for scheduling purposes. Queue management policies determine labor and machine utilization, manufacturing lead time, and priority control.

The relationship between sensitivity in MRP & CRP systems is a logical concern. Planned Orders, which are the output of MRP, serve as the primary input to CRP. This raises the issue of transferring instabilities from one system to the other. The heart of CRP is converting the time phased material requirements output from MRP into time phased labor and/or machine requirements. The capacity requirements are then loaded into weekly "buckets", or planning periods, for each work center. This total represents a number of different orders. As changes in individual orders occur, they tend to cancel one another out in terms of the weekly workload for any given work center. As a result, CRP sensitivity has never been considered a significant issue.

Recent simulation studies by Askin and Raghavan (1) have shown that the choice of a lot sizing strategy can effect the variability of the workload. They found that a fixed quantity policy, Economic Order Quantity (EOQ), caused the least disruption. A dynamic policy, Period Order Quantity, provided the greatest workload variability. They further observed that large batch sizes created greater variability than smaller batch sizes. The authors suggest that where the cost of changing production levels is significant, work center variability may be an issue. Their research raise another concern about the use of dynamic lot sizing rules.

FMS FOR FABRICATION & ASSEMBLY OPERATIONS

Most FMS implementations to date have been in fabrication and assembly operations. Furthermore, the FMS does not entirely replace the functional shop. The systems design problem of making effective use of FMS technology is twofold. First, there is the problem of trying to design the sequencing and buffering policies within the system. Second, there is the problem of developing policies to integrate the FMS into the larger plant operation. These issues will be discussed in this section.

A definition of FMS is given below, based upon that given in the Flexible Manufacturing Systems Handbook (5). An FMS is defined by three physical attributes:

1. A set (of more than one) numerically controlled (general purpose) machine tools.
2. Automatic material handling equipment linking the machine tools in the system together.
3. An overall computer control system that coordinates the machine tool operations and the material flow through the system.

There are a number of other system features that may be found in specific FMS implementations. For example, automatic tool changing (ATC), the use of robotics, adaptive control, and Automatic Storage and Retrieval Systems (ASRS) are frequently used in FMS systems.

Material flow in the FMS is highly flexible. The flow may be serial, parallel, or random, representing flow shop, parallel machine, and job shop structures. The logical view is determined by the control system's software. Material travels through the system on specially designed pallets. In the case of prismatic parts, the pallets act as fixtures for the work piece. For rotational parts, the pallet serves to position parts that will be placed in the chuck by a robot. Parts start as raw material in the form of a casting or bar stock. Prismatic parts make two passes through the system. The work piece is turned over and refixtured for the second pass to allow metal working on all faces. The distribution of operations between machines is largely a function of tooling requirements. A work piece may enter the system and be processed on a single machine, or be processed in a "random" sequence by several machines tools.

Work in process in the FMS consists mainly of a queue in the pallet setup area, at the gateway to the system. A second queue accumulates in the pallet

tear down area during unmanned operation. There is at least one pallet per machine tool, which provides the work piece currently being machined. Each machine tool may have room for an additional pallet on its pallet exchange mechanism. Some of the earlier implementations provided for a small queue of pallets at the individual machine tool. This was done by having a short spur for each machine when conveyor systems were used. In systems where vehicles were used for materials handling, a small carrousel for each machine served the same purpose. Queuing capability is needed at the individual work center when the materials handling system can not support unmanned operations with a common queue. As will be explained below, the level of work in process needed to support the FMS is approximately one tenth of that needed for conventional job shop operations.

FMS Design & System Buffers

The structure of the FMS is significantly different from that of the job shop, which is used in conventional machining. As a consequence, the approach to using buffers in the system design differs substantially. FMS functional design and the role of buffers in the system are discussed in this section.

Queues in the FMS are not needed, as they are in the job shop, to allow the machine tools to operate independently. There is a reduced need for coordination as a result of the machine tool's flexibility. For example, several operations, which would each require a separate routing step with conventional manufacturing, may be combined on a single machine tool. The coordination tasks that remain to be performed in the FMS are more manageable than they would be in the job shop, because the host computer is available. Queues function as a buffer in FMS only in the sense that they isolate the manual operations of pallet setup and tear down from the highly automated processes within the system. Queues of work in process are not needed as buffers for each machine in the system. The system's flexibility makes it possible to maintain a high utilization by flowing work from the setup area to the machines. The primary role of queues in the FMS is to allow unmanned operation for an eight or sixteen hour period.

Lead times using FMS technology are much less than those needed in conventional manufacturing systems. Material flow through the FMS is much faster than that through the job shop. Setup times are nearly eliminated as a result of palletized fixturing, the reduced number of machine tools needed to fabricate the part, and automatic tool changing. Travel time is rapid, using automatic conveyors, or automatic guided vehicles. Wait time for transit is near zero. Queue time within the system is much less than that of a job shop, with the greatest queue being in the pallet setup area. Total queue time in a FMS system is between 16 and 24 hours. In the job shop there is typically an allowance of two days per operation (machine tool). Since more machine tools are needed in a job shop, the total queue time planned for a part having five process steps would be 10 days. The result is nearly a ten fold decrease in both lead time and work in process.

Safety lead time is one of the buffering techniques used in MRP systems that is not needed in FMS systems. Uncertainty in timing is greatly reduced, in proportion to the reduction in lead times. This reduction occurs largely as a result of the decreased "window of vulnerability", or the more limited exposure to system uncertainties.

Tool management is always a factor in economical machining. However tool scheduling is not a critical factor in job shop operations. The velocity of the parts through the job shop is slow, and there are many opportunities for common setups. With the high velocity of parts through the FMS, tool scheduling is of much greater importance. The correct tool must be in the right place at the right time or processing will be suspended. The high capital investment in the FMS, and the greater part volumes processed, make downtime on a FMS much more expensive and disruptive than downtime on a conventional machine tool. Tool scheduling on an FMS must deal with more frequent changes in tooling, as a result of the flexibility required. The increase in tool wear brought about by the more intensive metal cutting also increases the frequency of tool changes. This problem becomes manageable by having an automatic tool changing capability. ATC designs typically have a capacity of 60 or more tools at a single machine. The FMS design problem, in terms of tool management, involves choosing the correct mix of tools to populate the tool magazine at each machine, and monitoring tool wear so as to make a timely replacements of worn tools.

Buffering in tool management is reflected in the capacity of the tool magazines for machine tools, and the degree of redundancy in each magazine. The tool buffering decision is part of the more fundamental FMS design decision of allocating cutting tools to the machine tools. Adequate tool buffering is essential to achieving the high machine utilization potential of FMS technology.

Materials handling is an integral part of FMS operations. The materials handling system's primary function is to move the product between the work stations in the FMS, including the machine tools, setup and tear down areas, washing, and inspection stations. There may also be a requirement to replace worn tools. Vehicle routing and scheduling presents a nontrivial problem. Provisions for vehicle safety, breakdown, and servicing must be included in the many systems that rely on AGVs, carts, or other vehicles. An acceptable solution must be found to the vehicle scheduling problem if the system is going to perform as expected.

The need for vehicles is a function of the number of work stations and the cycle times for the parts. The larger the number of work stations, and the shorter the cycle times the more vehicles needed. The demand for vehicle services also depends upon the product mix and lot sizing, the cutting tool allocation and life cycle, and vehicle maintenance needs such as battery charging, lubrication, inspection, and repair. The proper number of vehicles should be determined by simulation studies. Demands for vehicle service are stochastic in nature. It follows that the frequency and severity of delays waiting for vehicle service is a function of the number of vehicles in the system. Having an additional vehicle, to maintain high levels of vehicle service during vehicle breakdowns, is a "buffering" decision for the materials handling system.

Robot management involves managing robotic operations that transfer parts, tooling, and fixtures between the transportation system and the machine tool. Robots may also have to select the correct set of grippers for a given operation, and may play a critical role in automated inspection. Robots have their own controllers, which govern the detailed sequence of motions needed to fulfill their function. The host computer control system must coordinate

the robot by issuing the appropriate commands. Buffering decisions in the context of robot management relate to having spare units and spare parts.

Fixture management is pallet management in prismatic systems. In robotic systems it involves changing chucks and robot grippers. Pallets serve the dual role of positioning the part for materials handling, and positioning the part for machining. Therefore pallet positioning is done to close tolerances. The pallet transfer mechanism, used to exchange the pallet between the transportation system and the machine tool, must also hold close tolerances. Fixturing is done manually at the gateway to the system. General purpose pallets are designed to accommodate any fixtures needed for the part families processed on the FMS. The use of pallets in FMS has the effect of taking the fixturing part of the machine setup off-line.

The number of pallets and fixtures determine the capacity of the system to have work in process, or queuing, at the system gateway. This decision is related more to the period of time that the system is to be operated unmanned, than it is to buffering considerations.

Yield allowances in FMS, in terms of piece count, are minimal, if not altogether unnecessary. The precision in part fixturing, probing, and adaptive control capabilities of modern machine tools makes setup loss a thing of the past. Compensation is made for casting dimensions and positioning errors before machining begins. Modern machining practices minimize the chances of making a part out of tolerance. In-line inspection with coordinate measuring machines, provides feedback to the host computer, which in turn can make adjustments in the machine tools. The state of the art for adaptive control compensates for tool wear, thermal expansion of the part and the machine tool, and any dimensional irregularities of the machine tool itself. This new technology maintains close dimensional control during machining. While all modern machine tools do not have state of the art adaptive control, most have tool wear compensation and probing.

The computer and communications hardware for FMS contain a great deal of buffering in terms of redundancy, and spare parts. The host computer is usually in a duplex configuration, using an identical computer as a backup. A further discussion of the design of the computer and communications hardware is beyond the scope of this paper.

Integrating FMS And MRP

Conventional machining technology and FMS technology are as different as the job shop and the flow shop. This raises the question of how to manage the two technologies under the same roof. MRP has widely been accepted as the framework for managing conventional machining operations for part fabrication. Can the MRP system be modified to accommodate FMS technology ? If so, then what modifications are necessary ? The strategy offered in this paper is to treat the FMS as a single work center. A conceptual framework for integrating FMS technology into the existing MRP system will be presented in the sections that follow. The modifications necessary to implement this strategy will be briefly discussed.

Materials planning systems not effected by FMS technology will be those that are used to manage the end item. These systems include forecasting, aggregate

production planning, and the MPS.

The inventory system is essentially unchanged. As part of the FMS design process, component parts will be classified on the basis of processing similarities, using group technology concepts. If such information is not already in the item master, it should be appended. Various parameters in the Item master may also change. For example, yield losses will be virtually nonexistent, lead times will be greatly reduced, and minimum quantities will be one. Manufacturing cost for the item should be significantly reduced. If the same part is to be processed using both conventional machining and FMS technology it may be necessary to use separate part numbers

The bill of materials file will remain structurally the same. As with the inventory records, lead times will have to be adjusted.

MRP logic does not have to be changed to accommodate FMS. The main parameters that change are the lot sizing policies. With a zero setup cost, a lot for lot policy makes the most sense. Lead times should be no more than one week.

Product routings will need major revisions needed to accommodate FMS technology. Many of the routing steps used in conventional machining can be combined in the FMS. Many of the other processing parameter relating to setup time, run time, labor requirements, and yield will have to be changed as well.

Work center records will have to be appended to include a record for the FMS. This is the key to integrating the FMS into the existing materials management system.

Capacity planning and control will treat the FMS as it would any other work center. Resource requirements planning, if used, will include the FMS as a resource. In this way, FMS technology will indirectly affect the production plan. Input/output reports will be the same as in the existing system.

Priority control will function as in the conventional system in guiding the job to the FMS, and will help guide it downstream upon the completion of FMS operations.

Scheduling within the FMS will be handled independently by the FMS host. In concert with the MRP frame work, operation schedule dates will be a dominant factor in determining scheduling priority.

Activity reporting within the FMS system will be separate. It will be necessary to have a shop floor terminal in the pallet setup and tear down areas to report the general status of FMS operations to the MRP system. The host computer will track and log virtually every activity that occurs within the FMS. A separate management reporting system will be provided by the FMS host computer, or from files transferred to the main business computer.

CONCLUSIONS

1. The degree of coordination required in FMS is much greater than that required in conventional machining. The high velocity of jobs through the FMS requires close synchronization required between job, machine tool, cutting tool, pallet/fixture, and materials handling in order to realize high levels of machine performance. The large queues of work in process found in conventional machining systems reduces the need for close coordination in the job shop.

2. Adequate system buffers for FMS systems are essential in achieving the system design objectives. This follows directly from the first conclusion.

3. FMS technology drastically changes the role of queues of work in process. In an FMS system queues are needed to sustain unmanned operations, not serve as buffers for the individual machine tools. This fundamental change in the system structure makes it possible to reduce work in process inventories, and manufacturing lead times by an order of magnitude.

4. Safety lead time is not required for FMS products. This follows as a result of the reduction in the magnitude of lead time, and the corresponding reduction in the magnitude of lead time variability.

5. Safety lead time is inappropriate for component parts to be fabricated in house. It inflates lead time, and thus work in process. Safety lead time also undermines the formal system by diluting priorities.

6. Yield allowances, in terms of piece counts, are not needed for FMS machining. Set up scrap is eliminated as a result of the precision in fixturing, probing, tool wear compensation, and a reduction in the number of set ups needed. The dimensional control capabilities in FMS technology make process scrap, in terms of out-of-tolerance parts, a thing of the past.

7. A safety stock of component parts is not needed for FMS applications. The variability in timing and quantity for component production is minimal. Quality rejects for out-of-Tolerance work are minimal, eliminating a major source of unforeseen demand. In addition, there is a minimal set up cost and the ability to quickly respond to other sources of unforeseen demand.

8. Service parts, and spare parts for field failures, do not have to be inventoried for FMS applications. This follows directly from the argument given for the elimination of safety stock.

9. Lot-For-Lot is the appropriate lot sizing policy for FMS applications. The minimal setup costs make a lot size of one economically feasible. In an MRP setting, lot for lot will satisfy MRP requirements with a minimum work in process inventory.

10. Integrating FMS and MRP technologies can be accomplished with a minimal disruption by treating the FMS as a single work center. The major change needed to integrate the two systems is to revise the routing and item master records for those parts to be manufactured using FMS technology.

REFERENCES

1. Askin, Ronald G., and M. Raghavan, "The Effect of Lot-Sizing on Workload Variability," Journal of Operations Management, Vol. 4, No. 1, (November, 1983), pp. 53-71.

2. Carlson, R. C., J. V. Jucker, and D. H. Kropp, "Less Nervous MRP Systems: A Dynamic Economic Lot-Sizing Approach," Management Science, Vol. 25, (1979), pp. 754-761.

3. Collier, David A., "Research Issues for Multi-Level Lot Sizing in MRP Systems," Journal of Operations Management, Vol. 2, No. 2, (February, 1982), pp. 113-123.

4. Eichert, Edwin S., III, "Accounting for Unplanned Inventory Demands in Material Requirements Planning Systems," Production and Inventory Management, Vol. 15, No. 1, (1st Qtr. 1974), pp. 68-78.

5. Flexible Manufacturing Systems Handbook, Noyes Publications, Park Ridge, New Jersey, (1984).

6. Kanet, John J., "Toward Understanding Lead Times in MRP Systems," Production and Inventory Management, Vol. 23, No. 3, (3rd Qtr. 1982), pp. 1-14.

7. Kropp, Dean H., Robert C. Carlson, and James V. Jucker, "Use of Dynamic Lot-Sizing to Avoid Nervousness in Material Requirements Planning Systems," Production and Inventory Management, Vol. 20, No. 3, (3rd Qtr. 1979), pp. 49-58.

8. Kropp, Dean H., and Robert C. Carlson, "A Lot-Sizing Algorithm for Reducing Nervousness in MRP Systems," Management Science, Vol. 30, No. 2, (February, 1984), pp. 240-244.

9. Mather, Hal, "Reschedule the Reschedules You Just Rescheduled - Way of Life for MRP?," Production and Inventory Management, Vol. 18, No. 1, (1st Qtr., 1977), pp. 60-79.

10. Orlicky, Joseph, Material Requirements Planning, McGraw Hill, New York, (1975).

11. Steele, Daniel C., "The Nervous MRP System: How to do Battle," Production and Inventory Management, Vol. 16, No. 4, (4th Qtr. 1975), pp. 83-89.

12. Theisen, Earnest C., "New Game in Town - The MRP Lot Size," Production and Inventory Management, Vol. 15, No. 2, (2nd Qtr., 1974), pp. 1-13.

13. Wagner, Harvey M., and Thomas M. Whitin, "Dynamic Version of the Economic Lot Size Model," Management Science, Vol. 5, (1958), pp. 89-96.

14. Wemmerlov, Urban, "Design Factors in MRP Systems: A Limited Survey," Production and Inventory Management, Vol. 20, No. 4, (4th Qtr., 1979), pp. 15-35.

15. Whybark, D. Clay, and J. Gregg Williams, "Material Requirements Planning Under Uncertainty," Decision Sciences, Vol. 7, No. 4, (October 1976), pp. 595-606.

REAL-TIME OPTIMIZATION IN AUTOMATED MANUFACTURING FACILITIES

TUESDAY

8:00 REGISTRATION
8:45 Welcoming remarks

9:00 S. Gershwin, MIT: An approach to hierarchical production planning and scheduling.
9:30 P. O'Grady, NC State: A hierarchy of intelligent scheduling and control systems for automated manufacturing.
10:00 T. Baker, Chesapeake, Inc.: The integration of Planning, scheduling and control in automated manufacturing.

10:30 COFFEE BREAK

11:00 D. Liu, Hughes Aircraft: Intelligent manufacturing planning systems.
11:30 G. Chryssolouris, MIT: A decision-making framework for manufacturing systems.
12:00 S. Lawrence and T. Morton, Carnegie-Mellon: PATRIARCH--Hierchical production scheduling.
12:30 R. Conway and W. Maxwell, Cornell: A microcomputer system for real-time scheduling.

1:00 LUNCH

2:00 C. McMillan and F. Glover, Univ. of Colorado: A heuristic programming system for scheduling--an application of managerial robotics.
2:30 M. Fox, Carnegie Group, Inc.: Artificial intelligence approaches to factory scheduling and control.
3:00 P. Papas, Westinghouse, Inc.: ISIS--a project in review.

3:30 COFFEE BREAK

CONTRIBUTED PAPER SESSION 1

4:00 J. Maley, S. Ruiz-Mier, and J. Solberg, Purdue Univ.: Dynamic optimization in automated manufacturing--a knowledge integrated approach.
4:20 R. McPherson and K. White, Univ. Va.: A management control approach to the manufacturing, planning, and scheduling problem.
4:40 M. Steffen, Draper Labs, and T. Green, VPI: Hierarchies of sub-periods in constraint-directed scheduling.
5:00 M. Shaw, Univ. Illinois: A two-level planning and scheduling approach for computer-integrated manufacturing.

CONTRIBUTED PAPER SESSION 2

4:00 J. Bean and J. Birge, Univ. Mich.: Match-up real-time scheduling.

4:20 C. Chung, Univ. Kentucky: A maximal covering model for loading flexible manufacturing systems.

4:40 M. Langston and A. Morasch, Wash. State Univ.: Interstage transportation planning in the flow-shop environment.

5:00 L. Platzman and J. Bartholdi, Georgia Inst. Tech.: Minimal technology routing and scheduling systems based on space filling curves.

6:30 RECEPTION

WEDNESDAY

9:00 R. Wysk, S.D. Wu, and Y.S. Yang, Penn State: A multi-pass scheduler for flexible manufacturing systems.

9:30 R. Young, Texas A&M, and M. Rossi, Texas Instruments: Real-time scheduling for process operation activities in an FMS.

10:00 D. Pope, LTV Aerospace and Defense: Requirements for automatic control of aerospace manufacturing processes.

10:30 COFFEE BREAK

11:00 R. Rachamaduau, N. Raman, and B. Talbot, Univ. Mich.: Due-date based scheduling in a flexible manufacturing system.

11:30 J. Muckstadt, P. Jackson, and W. Maxwell, Cornell: Scheduling mixed-model jobs when operation times are identical.

12:00 R. Wittrock, IBM Manufacturing Research Center: An adaptive scheduling algorithm for flexible flow lines.

12:30 M. Meketon, AT&T Bell Labs: A production schedule model and algorithm for a flexible manufacturing line.

1:00 LUNCH

2:00 K. Stecke, Univ. Michigan: Determining aggregated FMS production ratios and minimum inventory requirements.

2:30 V. Chiodini, Honeywell Computer Sciences Center: An expert system for dynamic manufacturing rescheduling.

3:00 H. Watts, I.P. Sharp Associates, Ltd.: Dispatching--the critical automation link.

3:30 COFFEE BREAK

CONTRIBUTED PAPER SESSION 3

4:00 A. Kiran, USC, and S. Alptekin, Univ. Missouri: Scheduling jobs in flexible manufacturing systems.

4:20 F. Rodammer and K. White, Jr., Univ. Va.: Design requirements for a real-time production scheduling decision aid.

4:40 O. Maimon, Digital Eq. Corp.: Real-time operational control of flexible manufacturing systems.

5:00 M. Han, Texas A&M, and L. McGinnis, Georgia Inst. Tech.: Work flow control in automated manufacturing.

CONTRIBUTED PAPER SESSION 4

4:00 W. Davis, Univ. Illinois: Optimizing a Kanban production control system for a custom door manufacturer.

4:20 Y. Ho, Harvard, R. Suri, Univ. Wisc., and G. Diehl, Network Dynamics, Inc.: A technique for real-time sensitivity analysis of automated manufacturing systems.

4:40 A. Chandawarkar, VPI, and R. Wysk, Penn State: A linear programming approach to NC face milling.

5:00 W. Darrow, Towson State: System buffers required when FMS are used in fabrication and assembly operations.

Symposium on Real-Time Optimization
in Automated Manufacturing Facilities
Attendees

Sema Alptekin
University of Missouri, Rolla
Dept. of Engineering Management
Rolla, MO 65401

J. M. Anthonisse
Ctr. for Mathematics & Computer Sci.
Kurslaan 413, 1098 SJ Amsterdam
The Netherlands,

K. R. Baker
Dartmouth College
Amos Tuck School
Hanover, NH 03755

Thomas E. Baker
Chesapeake
P.O. Box 65
Berkeley Hts, NJ 07922

Keith E. Baldwin
NCSU - Industrial Engineering Dept.
P.O. Box 7906
Raleigh, NC 27695-7906

Han Bao
North Carolina State University
Dept. of IE, P.O. Box 7906
Raleigh, NC 27695

Randall A. Bartlett
General Motors
One Pontiac Plaza, Plant 8
Pontiac, MI 48035

George Batt
Grumman Data Systems
1000 Woodbury Rd.
Woodbury, NY 11707

James C. Bean
University of Michigan
272 IOE Bldg.
Ann Arbor, MI 48109

Elizabeth Billings
Fisher Controls International, Inc.
205 South Center St.
Marshalltown, Iowa 50158

John R. Birge
University of Michigan
1205 Beal; Ind. & Op. Eng. Dept.
Ann Arbor, MI 48109

Linda Birch
NC State University
Raleigh, NC 27607

Robert Blankenhorn
Grumman
1000 Woodbury Rd.
Woodbury, NY 11797

J. Carroll
GMC - Central Foundry Div.
P.O. Box 70
Defiance, OH 43512-0070

Michael F. Carter
General Motors Technical Center
30400 Mound Road
Warren, MI 48090-9015

Andrea Caulfield
CPC, GM
One Pontiac Plaza, Plant 8
Pontiac, MI 48053

Peter Chadzynski
DEC
24 Porter Road
Littleton, MA 01460

Aseem Chandawarkar
Virginia Polytechnic Institute
Dept. of IEOR
Blacksburg, VA 24061

Virginio Chiodini
Honeywell Computer Sciences Center
1000 Boone Ave., North
Golden Valley, MN 55427

Chen-Hua Chung
University of Kentucky
333 C. Commerce Bldg., Dept. of Mgmt.
Lexington, KY 40506-0341

Michael Clifford
Rensselaer Polytechnic Institute
Troy, NY 12180

Joel Cohen
Electronic Data Systems
755 West Big Beaver Rd., 8th Floor
Troy, MI 48084

Alan Cypher
Intellicorp
Mountain View, CA 94040

William P. Darrow
Towson State University
Dept. of Business Administration
Towson, MD 21204

Wayne J. Davis
Univ. of Illinois
104 S. Mathews Ave.
Urbana, IL 61801

Alan Desrochers
Rensselaer Polytechnic Institute
Troy, NY 12180

Kevin A. Dietrich
GM-CPC Pontiac Engin.
One Pontiac Plaza, Manhattan Proj.
Pontiac, MI 48053

Jasper B. Dowling
Martin Marietta Data Systems
Orlando, FL

Dana Duncan
Texas Instruments
Spring Creek
Plano, TX

John Ely
Martin Marietta Data Systems
103 Chesapeake Park Plaza MP #93
Baltimore, MD 21220

Edward Farnsworth
Digital Equipment
100 Minuteman Rd., MS 1/62
Andover, MA 01810

Charles A. Fausel
Electronic Data Systems
1805 Veterans Memorial Parkway
Saginaw, MI 48605-5073

Suellen Fausel
700 N. Johnson
Bay City, MI 48708

Steve Garrett
Theodore Barry & Assoc.
50 Rockefeller Plaza
New York, NY 10020

Kenneth Gehner
AT&T Eng. Research Center
P.O. Box 900
Princeton, NJ 08540

Stanley Gershwin
MIT
Bldg. 35-238, 77 Massachusetts Ave.
Cambridge, MA 02139

Fred Glover
University of Colorado
Applied Artificial Intelligence, Box 419
Boulder, CO 80309

Steven Gold
General Electric
P.O. Box 8555, Rm. 1954, Bldg. 9
Philadelphia, PA 19101

Richard R. Goth
Honeywell, Inc.
1200 E. San Bernardino Road
West Covina, CA 91790

Dan Grassetti
Data Checker/DTS
800 Central Expressway
Santa Clara, CA 95052-8112

Heinz Groeflin
AT&T Bell Laboratories
Crawfords Corner Road
Holmdel, NJ 07733

Stephen Grotzinger
IBM
T.J. Watson Res. Ctr., P.O. Box 218
Yorktown Heights, NY 10598

D. Gupta
University of Waterloo
Dept. of Management Sciences
Waterloo, ONT N2L 3G1

Gwen Hacker
Air Products & Chemicals, Inc.
P.O. Box 538
Allentown, PA 18105

Dan Hall
GMF Robotics
1410 Allen Drive
Troy, MI 48083

Walt A. Hasbach
Jet Propulsion Laboratory
4800 Oak Grove Dr., MS 170-104
Pasadena, CA 91109

Janie Helms
Texas Instruments
6500 Chase Oaks
Plaino, TX 75023

Kathleen L. Hendricks
General Electric
1000 Western Ave. MD 2103H
Lynn, MA 01910

Richard Hildebrant
C. S. Draper Lab
555 Technology Square
Cambridge, MA 02139

Melvin Hill
Martin Marietta Data Systems
103 Chesapeake Park Plaza MP #93
Baltimore, MD 21220

Gary Hood
Electronic Data Systems
755 West Big Beaver Rd., 8th Floor
Troy, MI 48084

Daniel Horn
Electronic Data Systems
755 West Big Beaver Rd., 8th Floor
Troy, MI 48084

Donald E. Hubicki
Watervliet Arsenal
SMCWV-ODP-S, Bldg. 25-3
Watervliet, NY 12189

Burton Hummer
Dupont
P.O. Box 27001
Richmond, VA 23261

G. K. Hutchinson
Univ. of Wisconsin-Milwaukee
540 Bolton Hall
Milwaukee, WI 53201

Ivan Johnson
C. S. Draper Lab
555 Technology Square
Cambridge, MA 02139

Pravin K. Johri
AT&T Bell Laboratories
Crawfords Corner Road, Room 4K420
Holmdel, NJ 07733

Reginald Joules
Martin Marietta Data Systems
98 Inverness, Suite 105
Englewood, CA 80112

Bob Joy
Northrop Corporation
One Northrop Ave. 5093/87
Hawthorne, CA 90250

Lindell J. Juergens
GM Tech. Ctr., Advanced Engin. Staff
30400 Mount Road, EA-MD767
Warren, MI 48090-9015

Ali S. Kiran
USC, Dept. of ISE
OHE 400, MC 1952
Los Angeles, CA 90089

Joannis A. Koskosidis
Princeton University
Dept. of Civil Eng., E-QUAD
Princeton, NJ 08544

Stephen R. Lawrence
Carnegie-Mellon Univ.
5000 Forbes Ave.
Pittsburgh, PA 15215

Kwan-Heng Lee
NCSU
Raleigh, NC 27650

William S. Liebert
Knoll Int.
Water St.
E. Greenville, PA 18041

Chin Wen Lin
Electronic Data Systems - Saturn
850 Stephenson Hwy, Suite 300
Troy, MI 48093

David Liu
Hughes Aircraft Company
P.O. Box 902, E4/M156
El Segundo, CA 90245

Sheldon Lou
MIT
77 Massachusetts Ave.
Cambridge, MA 02139

Gary L. Lucas
Decision-Science Applications
1901 N. Moore St.
Arlington, VA 22152

Hanan Luss
AT&T Bell Laboratories
Crawfords Corner Road, Rm. HO-46423
Holmdel, NJ 07733

Oded Maimon
Digital & MIT
146 Main St. (MLOS-2/E50)
Maynard, MA 01754

James G. Maley
Purdue University
Potter Bldg 114
W. Lafayette, IN 47907

Robert M. Martin
CPC, GM
One Pontiac Plaza, Plant 8
Pontiac, MI 48053

Rex Martin
Texas Instruments
6500 Chase Oaks
Plaino, TX 75023

William Maxwell
Cornell University
OR & IE, Upson Hall
Ithaca, NY 14853

Edward D. McDowell
Oregon State University
Industrial Engineering
Corvallis, OR 97331

Leon F. McGinnis
Georgia Tech.
Atlanta, GA 30332

Marc Meketon
AT&T Bell Laboratories
Holmdel, NJ 07733

Unny Menon
California Polytechnic State Univ.
Industrial Engineering Dept.
San Luis Obispo, CA 93407

Pitu B. Mirchandani
Rensselaer Polytechnic Institute
Troy, NY 12180

Daniel Monahan
NC State Univ.
Raleigh, NC 27650

John Moore
University of Waterloo
Management Sciences Dept.
Waterloo, Ontario, Canad

Annette Morasch
Hewlett-Packard, McMinnville Div.
1700 S. Baker St.
McMinnville, OR 97128

Philip D. Motz
Cincinnati Milacron
4701 Marburg Ave., Dept. 61M
Cincinnati, OH 45209

Henry L. Nuttle
NC State University
Dept. of Ind. Eng., Box 7906
Raleigh, NC 27695

Peter O'Grady
North Carolina State University
Dept. of Industrial Engineering
Raleigh, NC 27695

Nicholas G. Odrey
Lehigh University
Dept. of IE, Packard Lab #19
Bethlehem, PA 18015

Mufit Ozden
Miami University
Systems Analysis Dept.
Oxford, OH 45056

Petros N. Papas
Westinghouse Electric Corp.
P.O. Box 160
Pittsburgh, PA 15230

George T. Patton
Patton Associates
3608 S. 12th St.
Arlington, VA 22204

Doris Pavonarus
Knoll Int. Inc.
Data Base Sup.
East Greenville, PA 18041

Michael L. Pinedo
Columbia University
JEOR Dept.
New York, NY 10027

Loren Platzman
ISYE/GA Tech
Atlanta, GA 30332

Charlie Poe
Texas Instruments
6500 Chase Oaks
Plaino, TX 75023

Behnam Pourbabai
New York University
Stat/OR Dept., 100 Trinity Place
New York, NY 10006

George E. Pugh
Decision Science Applications, Inc.
1901 N. Moore
Arlington, VA 22209

Rick A. Ross
C-P-C-Pontiac
One Pontiac Plaza-Plant 8
Pontiac, MI 48053

Mike Rossi
Texas Instruments
1235 Trinity Mills
Carrollton, TX 75006

Sergio Ruiz-Mier
Purdue University
Center for Intelligent Manu. Systems
West Lafayette, IN 47907

Raymond J. Ryan
General Motors
77 W. Center St.
Saginaw, MI 48603

Randhawa Sabah
Oregon State University
Industrial Engineering
Corvallis, OR 97331

Ezat T. Sanii
N.C. State University
Dept. of IE. Box 7906
Raleigh, NC 27695

Arvind Sathi
Carnegie Group Inc.
650 Commerce Ct., Station SQ
Pittsburgh, PA 15219

Winifred Schmidt
Karlsruhe Nuclear Research Center
1 Farragut Square So.
Washington, DC 20006

Ralph Seeley
USDA
1400 S. Joyce St., Apt. A608
Arlington, VA 22202

Moshe Segal
AT&T Bell Labs
Room 4M-317
Holmdel, NJ 07724

Ali Sharifnia
Boston University
College of Engin., 110 Cummington St.
Boston, MA 02215

Mike Shaw
University of Illinois
350 Commerce West
Champaign, IL 61820

Paul Shuttleworth
Knoll International Inc.
Water St.
East Greenville, PA 18041

Hugo Simao
Princeton University
Dept. of Civil Eng., E-QUAD
Princeton, NJ 08544

Rick So
AT&T
P.O. Box 900
Princeton, NJ 08540

Mitchell S. Steffen
C.S. Draper Lab
MS 2C, 555 Technology Square
Cambridge, MA 02139

Richard Tracey
Saturn Corporation
1400 Stevenson Highway
Troy, MI 48007

Don C. Trujillo
Jet Propulsion Laboratory
4800 Oak Grove Dr., MS 170-104
Pasadena, CA 91109

W. J. Trybula
General Electric Company
P.O. Box 8106
Charlottesville, VA 22906

Y. Y. Tsang
Electronic Data Systems
755 West Big Beaver Rd., 8 Floor
Troy, MI 48084

John J. Turner
DuPont Co.
Bldg. 302, Exp. Sta.
Wilmington, DE 19898

Hiten Varia
Electronic Data Systems
755 West Big Beaver Rd., 8th Floor
Troy, MI 48084

Henry Watts
I.P. Sharp Associates
4699 Old Ironsides Drive, Suite 300
Santa Clara, CA 95054

K. Preston White, Jr.
University of Virginia
Systems Engineering, Thornton Hall
Charlottesville, VA 22901

Cynthia K. Whitney
C. S. Draper Lab
555 Technology Square
Cambridge, MA 02139

George R. Wilson
Lehigh University
Ind. Engr. Dept., Packard Lab #19
Bethlehem, PA 18015

Robert Wittrock
IBM Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Barry Wolf
Xerox Corporation
141 Webber Ave.
North Tarrytown, NY 10591

Robert E. Young
Texas A&M University
Dept. of Industrial Eng.
College Station, TX 77843

Jiri Zdarek
Czech Embassy
3900 Linnean Ave., NW
Washington, DC 20008

Dennis Zmolek
Fisher Controls Int.
205 S. Center St.
Marshalltown, IA 50158

NBS-114A (REV. 2-8C)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET (See instructions)	1. PUBLICATION OR REPORT NO. NBS/SP-724	2. Performing Organ. Report No.	3. Publication Date September 1986

4. TITLE AND SUBTITLE

Real-Time Optimization in Automated Manufacturing Facilities

5. AUTHOR(S)

Richard H.F. Jackson and Albert W.T. Jones, Editors

6. PERFORMING ORGANIZATION (If joint or other than NBS, see instructions)

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
~~WASHINGTON, DC 20234~~
Gaithersburg, MD 20899

7. Contract/Grant No.

8. Type of Report & Period Covered

Final

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (Street, City, State, ZIP)

U.S. Department of the Navy
Manufacturing Technology Program
Washington, DC 20360

10. SUPPLEMENTARY NOTES

Library of Congress Catalog Card Number: 86-600580

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)

The Symposium on Real-Time Optimization in Automated Manufacturing Facilities was held at the National Bureau of Standards, Gaithersburg, Maryland, January 21-22, 1986. It was jointly sponsored by the Center for Manufacturing Engineering (with funds obtained from the Navy Manufacturing Technology Program) and the Center for Applied Mathematics. It was designed to bring together those who design and test optimization procedures for solving planning, scheduling, and routing problems with those who must use these procedures in a real-time manufacturing environment. Included in the proceedings are discussions of the following topics: an approach to hierarchical production planning and scheduling; a hierarchy of intelligent scheduling and control for automated manufacturing systems; the integration of planning scheduling, and control for automated manufacturing; intelligent manufacturing planning systems; a decision making framework for manufacturing systems; PATRIARCH -- hierarchical production scheduling; low-level interactive scheduling; the general employee scheduling problem: an effective large scale solution approach; ISIS project in review; dynamic control in automated manufacturing: a knowledge integrated approach; a management control approach to the manufacturing, planning, and scheduling problem; hierarchies of sub-periods in constraint-directed scheduling; a two-level planning and scheduling approach for computer integrated manufacturing; match-up real-time scheduling; a maximal covering model for loading flexible manufacturing systems; interstage transportation planning in the flow-shop environment; minimal technology routing and scheduling systems based on space filling curves; a multi-pass expert control system (MPECS) for flexible manufacturing systems; requirements for automatic control of aerospace manufacturing processes; real-time scheduling of an automated manufacturing center; the interaction between design and scheduling in repetitive manufacturing environments; an adaptable scheduling algorithm for flexible flow lines (abstract); determining aggregated FMS production ratios and minimum inventory requirements; a knowledge based system for dynamic manufacturing replanning; dispatching -- the critical automation link; scheduling jobs in flexible manufacturing systems; design requirements for a real-time production scheduling decision aid; an optimized Kanban system for a custom door manufacturer; a linear programming approach to numerically controlled face milling; and, system buffers required when FMS are used in fabrication and assembly operations.

12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)

automated manufacturing, flexible manufacturing, hierarchical control, planning problems, routing problems, real-time optimization, scheduling problems.

13. AVAILABILITY

☒ Unlimited

☐ For Official Distribution. Do Not Release to NTIS

☒ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.

☐ Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES

481

15. Price

*U.S. GOVERNMENT PRINTING OFFICE: 1986-491-097:52603

NBS Technical Publications

Periodical

Journal of Research—The Journal of Research of the National Bureau of Standards reports NBS research and development in those disciplines of the physical and engineering sciences in which the Bureau is active. These include physics, chemistry, engineering, mathematics, and computer sciences. Papers cover a broad range of subjects, with major emphasis on measurement methodology and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Bureau's technical and scientific programs. Issued six times a year.

Nonperiodicals

Monographs—Major contributions to the technical literature on various subjects related to the Bureau's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NBS, NBS annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

Applied Mathematics Series—Mathematical tables, manuals, and studies of special interest to physicists, engineers, chemists, biologists, mathematicians, computer programmers, and others engaged in scientific and technical work.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a worldwide program coordinated by NBS under the authority of the National Standard Data Act (Public Law 90-396).
NOTE: The Journal of Physical and Chemical Reference Data (JPCRD) is published quarterly for NBS by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements are available from ACS, 1155 Sixteenth St., NW, Washington, DC 20056.

Building Science Series—Disseminates technical information developed at the Bureau on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NBS under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The standards establish nationally recognized requirements for products, and provide all concerned interests with a basis for common understanding of the characteristics of the products. NBS administers this program as a supplement to the activities of the private sector standardizing organizations.

Consumer Information Series—Practical information, based on NBS research and experience, covering areas of interest to the consumer. Easily understandable language and illustrations provide useful background knowledge for shopping in today's technological marketplace.
Order the **above** *NBS publications from: Superintendent of Documents, Government Printing Office, Washington, DC 20402.*
Order the **following** *NBS publications—FIPS and NBSIR's—from the National Technical Information Service, Springfield, VA 22161.*

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. The Register serves as the official source of information in the Federal Government regarding standards issued by NBS pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NBS Interagency Reports (NBSIR)—A special series of interim or final reports on work performed by NBS for outside sponsors (both government and non-government). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Service, Springfield, VA 22161, in paper copy or microfiche form.

U.S. Department of Commerce
National Bureau of Standards
Gaithersburg, MD 20899