JOURNAL OF OPERATIONS MANAGEMENT
Vol. 4, No. 2, February 1984

A Heuristic Programming Approach to the Employee Scheduling Problem And Some Thoughts On "Managerial Robots"

FRED GLOVER*
CLAUDE MCMILLAN*
RANDY GLOVER**

EXECUTIVE SUMMARY

We describe a system for the automatic scheduling of employees in the particular setting in which: the number of employees wanted on duty throughout the week fluctuates; the availabilities of the employees varies and changes from week to week; and a new schedule must be produced each week, by virtue of the changing demand for service.

The problem which we address appears in a variety of settings, including: airline reservation offices; telephone offices; supermarkets; fast food restaurants; banks and hotels.

Previous approaches to the problem have relied chiefly on formal methods, generally involving one or another variation of linear or integer, mathematical programming. We suggest that except in cases involving very small problems (only a handful of employees) that those approaches have not proven promising, especially where union rules and management requirements impose complex constraints on the problem, and that a heuristic approach has proven to be substantially superior.

We set forth the general features of our heuristic approach, which we see as an application of artificial intelligence; we show how, in contrast to other approaches, which design shifts as if employees were always available and try to fit those shifts to employees who are not always available, our system design shifts with deference to the employees' limited availabilities; we suggest that, for a given service level, our system produces schedules with a better "fit"—number of employees actually on duty comparing more favorably with the number wanted; and we state that while, for a given service level, a 'manual scheduler' may take up to 8 hours each week to prepare a good schedule, our system, on most micro computers, routinely produces better schedules involving up to 100 employees in about 20 minutes.

The scheduling of employees is generally considered to be a managerial function, in the setting of the problem we address. When a craft employee is replaced on an assembly line by a machine which performs the same function, we speak of the replacing mechanism as an industrial robot.

We suggest that systems like that which we describe deserve a name, to distinguish them from comparable, computer based systems which do not replace, but rather supplement a manager, and we suggest the name 'managerial robot' for such systems.

* University of Colorado at Boulder, Boulder, Colorado.
** Management Science Software Systems, Boulder, Colorado.

We set forth the characteristics which we feel would justify applying the term 'managerial robot' to a computer based system, and suggest that classification is basic to understanding and communication and that just as terms such as decision support systems and expert systems prove useful in our increasingly advanced, technological society, so also the term managerial robot has a place in our scheme of things.

Decision support systems do not qualify as managerial robots for the reason that managerial robots don't simply support the decision making process, but rather replace the manager in his performance of a function which, when performed by a human being, is considered a managerial function.

Nor do we consider managerial robots to qualify as expert systems. While our scheduling system contains an inference mechanism, and could be enhanced to improve the quality of its schedules thru 'experience' (and thus to 'learn'?), that—lacking a knowledge base in the sense of expert systems—and most of all in replacing rather than supporting the decision maker, the managerial robot needs a term of its own.

We elaborate, in this paper, a specific application of our system, and show how the design of shifts, and the placement of breaks, serve to yield a fit whose quality no human scheduler can duplicate.

THE EMPLOYEE SCHEDULING PROBLEM

Heuristic programming, as an approach to artificial intelligence, uses machines to solve intellectually difficult problems. One heuristic programming approach equips the machine to approximate procedures human problem solvers employ. Another equips the machine to employ methods which may be quite unlike those commonly employed by human beings, and which produce better solutions than human beings can produce and in much less time.

The heuristic programming approach we describe in this paper solves a problem faced by operational level managers in supermarkets, telephone offices, discount stores, fast food restaurants, airline reservation offices, banks and hotels. The problem calls for a management plan, specifically a schedule which assigns work shifts to employees who are more or less interchangeable. For reasons that will be described, our computer-based system for solving the problem qualifies as a "managerial robot"—a label we propose should be applied to systems that exhibit certain distinguishing characteristics, in order to differentiate them from systems that fulfill other currently defined roles, such as "decision support."

Since the mid-1970's certain parts (or features) of the problem we address have been recognized for their importance and discussed in the literature. For example, Henderson and Berry [4] addressed the problem of selecting a near optimal set of shifts, in the scheduling of telephone operators. Fluctuating demand and the placement of breaks within shifts were considered, but the varying availabilities of individual employees was not considered.

Krajewski and Ritzman [6] addressed the problem of determining a fixed weekly shift schedule for the encoder department in a large bank, utilizing any mix of full and part time employees. Theirs was an LP approach to determine the optimal number of full time and part time encoder clerks to assign to each of a number of preselected tour shifts. They did not include the positioning of breaks, nor the varying availabilities of individual employees.

Mabert and Watts [7] described the use of simulation to help produce a set of preselected tour shifts, with deference to general worker convenience, but again not with deference to the varying availabilities of individual employees.

We address the larger problem, in which varying employee availabilities are considered; minimum and maximum hours per day and per week for individual employees are con-

sidered for both full and part time employees; and union/management rules regarding start times and quantity of work, depending on the seniority of the employee, are respected.

More importantly, rather than working from a fixed set of tour shifts, in our approach shifts are designed as the scheduling progresses, with deference to employees' availabilities as well as to the manning requirements. As a result, a better fit is achieved between "actually assigned," by quarter hour periods throughout the week, and "targeted," with a smaller payroll.

The system we produced also assures that not only will the number of employees wanted on duty be closely approximated by the number assigned, by quarter hour periods, but also that a specified skill mix will be present.

The specifics of the problem we address vary from one setting to another, but the basic problem may be described as the following:

How to design shifts and assign them to employees for the week so that the payroll is small, while matching as closely as possible the number of employees actually on duty to the number wanted on duty, subject to:

1) Fluctuating requirements: At 7:00 am only a few employees may be required. As the day wears on more are needed, then the requirement decreases toward late evening. Tomorrow's requirements, also varying, may follow a different pattern. Furthermore, new week's requirements are generally assumed to be different from last week's;
2) Employee availabilities: Many employees are available for work only on certain days of the week and during certain hours on those days, some preferring part time work. Furthermore, their availabilities next week may not be the same as those this week;
3) Union and management rules which govern the design of the schedule: Rules and policies may severely limit the manager's freedom to compose and assign shifts. In addition, quarter hour breaks and lunch periods may be called for within specified time frames, and management must schedule employees to cover each other's breaks;
4) Continuity of employment: Some reasonable mix of full time and part time employees is generally called for. Moreover, the supervisor must spread the work among employees to provide some minimal level of employment continuity from week to week.

The problem becomes more complex when employees are not fully interchangeable—where those on duty must satisfy the requirements for a specific job skill mix. The system we describe makes that accommodation, but the application which we describe in what follows assumes interchangeability of employees.

SHORTAGES, OVERAGES AND THE SIZE OF THE PAYROLL

A schedule which fails to cover the requirements risks poor customer service and reduced revenue from inability to meet demand. A schedule which has more employees than needed to cover the requirements yields an inflated payroll.

In a "perfect" schedule requirements are met exactly and breaks are covered perfectly. But such schedules are rare. Manual schedulers generally aim for a close approximation of people on duty to people required. If shortages are more unattractive than overages the supervisor will accept a larger payroll to reduce the shortages to some acceptable level, particularly in periods during the week in which shortages are especially undesirable.

The stakes are high, and the uncommon supervisor who is highly skilled at this task is a valuable asset. Acquiring such a level of skill, however, takes not only a natural facility but months or even years of experience. Unfortunately, even highly skilled supervisors

have their "off days," and schedules produced at such times can be costly. Further, time and energy devoted to scheduling decreases the amount of these two resources that the supervisor can devote to other concerns that demand attention.

AN AUTOMATIC SCHEDULER

We have developed a versatile, user friendly software system for dealing with the problem described above. It performs the management planning function of an operational level manager, assigning work periods to employees throughout the week.

We will subsequently describe a simple example problem and indicate the solution obtained by our automatic scheduler, to provide a concrete illustration. Even at this level of simplicity, the problem—which was taken from a real world setting—requires of an experienced manager about 8 hours each week to prepare a schedule manually. Our system produces a superior schedule, reducing the total time to obtain it by more than 95%. This is accomplished without relying on a large "number crunching" computer, but by using a small and inexpensive microcomputer in the Z80 class, with floppy disk drives.

In terms of solution quality, our system more perfectly matches the number of employees on duty throughout the week with those wanted on duty, and does that with a significantly smaller employee payroll. As problem complexity increases, the system produces schedules of relatively higher solution quality by comparison to those generated by a skilled supervisor. Interactive components in the system are called into play to update the employee availability file and the requirements forecast, and to set parameters where appropriate to change the minimum and maximum shift duration and other dimensions of the problem. But once the preparation of the schedule begins, no human intervention is required.

We first provide an overview of the methodology by which our system generates such solutions, and then present the example problem and its solution.

THE HEURISTIC SOLUTION PROCEDURE

Heuristic programming methods characteristically have several components:

1) Criteria for defining a "solution," and for comparing two solutions on the basis of relative admissibility and desirability. (These criteria are not always transitive, due to difficulties of obtaining global evaluations—a phenomenon that can lead to "cycling.") A refined method may modify its criteria systematically at different solution stages, and, in particular, apply different criteria to final solutions (obtained at the end of a solution pass) than to intermediate solutions;
2) Rules that define the nature of a "move" by which one solution state transitions to another;
3) Criteria for differentiating relative admissibility and desirability of moves, generally translated into a single scale that measures a "composite" form of relative desirability. (Sometimes such criteria are maintained as a series of thresholds, each of which must be passed to allow a move to acquire the status of "admissible"—and, by extension, "desirable".) The criteria applicable to moves may not, in general are not, the same as the criteria applicable to the solution states to which they lead. The reason for this is that it may be very costly to identify the complete form of the solution that will ultimately result from a particular move, and hence the evaluation of alternative moves would slow to a standstill if such identification were required;

4) Rules for generating a subset of candidate moves to evaluate. Often, the number of possible moves by which one solution state may transition to others is exceedingly large, and it would be inordinately time consuming to evaluate them all. (One may think of chess as an example.) Instead some sort of sampling technique or screening device is used, to keep the moves examined to a manageable number;
5) Rules for reversing or revising previous moves. Because a solution state may evaluate differently than the move that led to it, sometimes a high ranking move can produce a low ranking outcome. When this occurs, it may be better to backtrack at once and select an alternative than to attempt to "dig out of the hole" by proceeding in the usual fashion;
6) Rules for combining or merging moves. Some procedures attempt to evaluate the outcome of making two (or even more) moves in sequence. This is because a state that "looks good" on the basis of a one-move analysis may in fact be poor—as when all moves available to succeed the first are inferior. A slightly less attractive first move may allow a more attractive successor. Combined moves are exceedingly expensive to evaluate (the number of possibilities to examine grows combinatorically) and methods that employ combined moves must be extremely well managed to avoid consuming massive amounts of time;
7) Rules for initiating the method, and for restarting the method. A method may undertake to "restart"—i.e., to execute more than one solution pass—depending on the length of time a single pass consumes and on the empirical likelihood of finding a better solution.

The preceding general characteristics of heuristic procedures provide a backdrop for describing the methods used by our automatic scheduler. We characterize these methods in a form that applies equally to the more complex case where the employees are not interchangeable.

As a basic for defining what we mean by a solution, we first define a tour (tour of duty) to consist of an assignment of working hours, together with lunch period and breaks (if applicable), for a given person on a given day. That is, a tour is generally a member of a range of possible working hour assignments, for a particular person and day (though other employees may have the same range of possibilities). A solution, then, is defined for our purposes as any collection of tours applicable to the week (or other span of time) under consideration.

A solution is locally admissible (relative to an intermediate solution stage) if it does not assign any employee more than his or her maximum weekly number of hours, while satisfying individual restrictions, union rules and company policy concerning required number and spacing of days off, and concerning minimum hours of break between successive days. A solution is globally admissible (relative to a terminal solution stage) if in addition it satisfies all other individual, union and company rules concerning minimum number of hours worked, the relative composition of part and full time employees, preferential working time slots for employees with greater seniority, and so forth.

A globally admissible solution is considered more desirable than another if it achieves a better fit of hours worked to the required hours on duty, and reduces the total payroll expenditure. (This may occur, for example, by reducing cases where hours on duty exceed the various pay categories, within limits stipulated by union regulations.) Each company attaches its own importance to such things as the number of hours short and over targeted

"requirement" levels, and the number or mix of employees working. Relative desirability of two globally admissible solutions must therefore be "defined" by the method to meet the company's own conception as closely as possible. This is accomplished by assigning appropriate weights to shortages, surpluses, and cost of labor, and combining these elements in a composite objective function. (The issue of appropriate weights can be nontrivial. In more difficult cases the issue is settled by feedback sessions with the user. Alternative weights are tested until those which yield the most desirable results from the user's viewpoint are identified.)

We distinguish the relative desirability of two intermediate (as opposed to "final") solutions on a specially restricted basis, allowing comparisons to be made only between solutions generated from a single (current) solution by a particular type of move.

To describe our criteria for evaluating relative desirability, we first characterize the types of moves we treat as fundamental:

1) Adding a tour (for a particular person on a particular day);
2) Deleting a tour;
3) Modifying a tour by changing its lunch period or breaks;
4) Exchanging a tour with another (for the same person and day);
5) Trading tours between two people on a given day;
6) Shifting a tour from one day to another (for the same person);
7) "Cross-trading" tours between two different people on two different days.

The more complex of the moves can of course be created from the simpler moves, and therefore may be viewed as selected types of 'combined' moves. In addition, the simple move (1) can be viewed as composed of two steps, the first selecting the starting and ending periods for the tour, and the second inserting the lunch and breaks into the tour.

Different types of moves are applied at different stages. At first, the method predominantly considers moves only of type 1, then gradually allows increasing numbers of moves of types 2 and 3, finally incorporating the remaining types of moves, using each type until no further improvement results.

"Improvement" is a derivative term linked to the relative desirability of intermediate solutions, whose "practical meaning" we now undertake to indicate. Because different companies have different criteria, the precise definition of relative desirability depends somewhat on the setting. However, in general, our method creates a weighting function for each component of global admissibility, as well as for the components of labor cost and "fit" (matching hours worked to requirements). Local, as opposed to global, admissibility is always maintained, and therefore is not incorporated into these functions. These weighting functions are combined so that in early stages, global admissibility is treated as a minor component of the whole—in effect, neglected except for the rules that dictate minimum spacing of duty tours. As the schedule begins to "fill up," these admissibility considerations become more pressing and those that are violated take on more weight. The more advanced types of solution moves are specifically used to move closer to satisfying global admissibility requirements, while still giving appropriate weight to the other components of relative desirability.

Note in particular the philosophy that particular types of moves and particular types of evaluations are best applied at different solution stages, though there is also a cycling through alternatives. (In our setting, this type of approach automatically accomplishes the

result of restricting attention to a manageable subset of candidate moves at any particular juncture.) An analogy with chess playing is prompted, where different strategies are most suitable to beginning, middle and end games. As in chess, our conclusions are based both on logical analysis and a good deal of experimentation, verifying empirically the approaches that work best. Unlike chess, however, our rules must vary a bit from game to game. Most significantly, we have an advantage no player of chess possesses, which invites the use of a very different and powerful type of strategy.

This strategy is based on the fact that it is possible, in this type of setting, to make a move that allows the solution quality to deteriorate, without "losing the game." In fact, creating the right kind of deterioration may set the stage for subsequent improvement that outweighs the initial setback. Allowing assignments (or moves) to oscillate between deterioration and improvement in a suitably controlled way is the key to making the approach effective. The basic ideas of this "oscillating assignment" type of strategy are given in [2], where a detailed illustration is provided of how the strategy is played in the setting of a discrete optimization problem.

The context of our automatic scheduler has necessitated additional refinements. Due to different criteria of desirability (and admissibility) by different users, we have designed the procedure to make its deteriorating moves first along the dimensions where the impact of deterioration is least significant. Subsequent "recovery moves" focus on making gains on the more important dimensions. There are never any guarantees that deterioration will be more than offset by ensuing gains, but the empirical success of the automatic scheduler attests to the usefulness of this approach.

The oscillating assignment approach subsumes the more customary type of "move reversal" approach as a special case, in addition, the approach provides an alternative to the expensive strategy of "combining moves." (However, as already indicated, we employ certain moves that may be viewed as special types of combined moves.) Our approach may be interpreted as seeking to uncover multi-move combinations that yield a net overall improvement without going through the extreme effort of composing (and calculating the full effects of) these combined moves in advance. Finally, oscillating assignment also somewhat achieves the effect of "restarting" the solution process, but without the large element of randomness usually employed (and typically sought) in such re-starts. By allowing controlled deterioration, the approach permits the solution to break away from the region to which it would otherwise be confined, and to work its way back toward improvement in a region that may be entirely different. The expense of dismantling the current solution and starting essentially from scratch, as in most restart approaches, is never incurred.

These general remarks about our strategy do not attempt to convey its nature in minute detail, in keeping with our goal of elucidating central concepts rather than providing a "cookbook" summary. Nevertheless, we do not wish to imply that the treatment of details is inconsequential for effective implementation.

We have saved to the last the description of one additional strategy employed by our automatic scheduler. All previous criteria for evaluation have been based on elements of admissibility and desirability directly meaningful to the goals a particular company wishes to emphasize. However, particularly in the early stages of the solution process, we employ a patterning criterion, with no immediately apparent connection to such goals. The requirements that tour assignments attempt to fit may be viewed as a bar graph, where the number of employees "needed" on duty in any time slot is represented by a vertical bar.

If these bars exhibit an irregular pattern, rather than one smoothly rising, descending, or uniform, the patterning criterion biases the selection of moves in favor of those that leaves the pattern of bars smoother after the move is made. Use of this criterion often seems to lead to good solutions after fewer total moves.

We have not of course touched upon the "systems analysis" features of the automatic scheduler, such as the design of data structures and the organization of operations within the computer. While important, such matters are treated elsewhere (see, e.g. [3]), and represent a level of implementation detail below that of the broader issues at the focus of this paper.

EXAMPLE PROBLEM—A CONCRETE ILLUSTRATION

We now present an example of a specific problem arising from a real world application and the solution our system generates for it. Some of the messier complications found in more difficult problems are not present in this application, making it more suitable for illustration purposes.

In Figures 1 through 4 sample data from one run of the system is displayed, in a setting where 25 employees were available for work.

Figure 1 shows graphically the hours of availability of the employees (on Sunday only). Thus employee #2 can start work as early as 07:00 on Sunday, must quit by 16:00, and is available for 40 hours during the week. Employee #12, on the other hand, can work any time on Sunday, but is available for only 31 hours during the week. Employees 3, 6 and others are unavailable anytime on Sunday.

In Figure 2 the hours assigned to the employees for the full week are shown in tabular form, and shown again for Sunday only in Figure 3, along with the timing of the breaks.

In Figure 4 the schedule for Sunday is shown graphically. Observe that employee #2 starts at 07:00, takes a quarter hour break at 08:45, a half hour break for lunch at 10:30, a second quarter hour break at 13:45 and quits at 15:30—an 8 hour shift with an unpaid half hour for lunch.

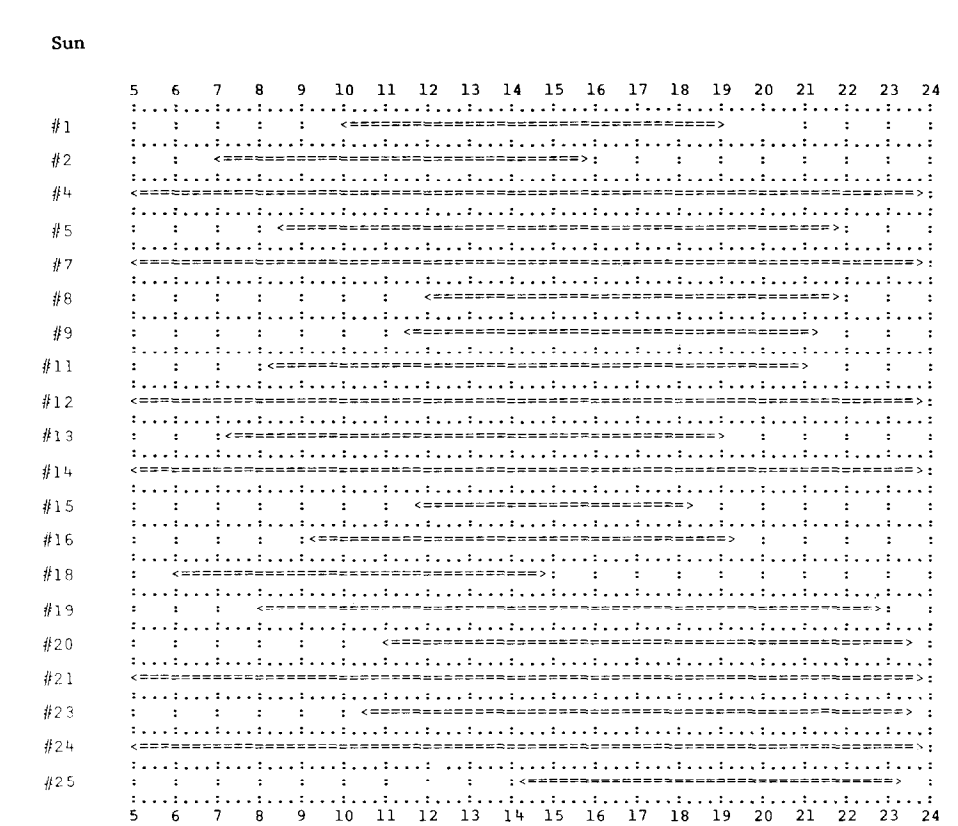
In this problem only full timers were entitled to 40 hours of work during the week, and part timers to a minimum of 16 hours for the week, with each shift ranging from 4 to 8 hours.

Note that at 14:45 (Fig. 4) ten employees are on duty but #8 is on his first quarter hour break and #11 is taking his lunch break, leaving "on the job" the 8 employees called for in the "required" row toward the bottom of Figure 4. Notice also the pattern of requirements, starting with 1 employee at 07:00, rising to a maximum of 10 at 14:15 and fluctuating throughout the day.

For Sunday our schedule came up "short" 1 employee in each of 2 quarter hour periods (18:30 and 18:45), and "over" 1 employee in 5 quarter hour periods—a match between "employees wanted" and "employees on duty" significantly better than a human scheduler can deliver manually, while complementing it with assignments for the rest of the week and meeting the various union/management rules.

As stated earlier, an experienced manager requires about 8 hours each week to manually prepare a schedule like that portrayed in Figures 1 through 4. Our system produces a schedule like that on a small microcomputer in the Z80 class, with floppy disk drives, in about 20 minutes. Our system more perfectly matches the number of employees on duty throughout the week with those wanted on duty, and does that with a significantly smaller employee payroll.

FIGURE 1



THE AUTOMATIC SCHEDULER AS A "MANAGERIAL ROBOT"

We have previously suggested that an attempt to classify our system might appropriately use the descriptive label "managerial robot," a term we propose by analogy to the more common term "industrial robot."

Industrial robots catch the popular fancy partly because they appear vaguely human as they reach, grasp, position and sometimes assemble objects.

However, they qualify as robots not because of their appearance but because of the functions they perform, functions which would otherwise be performed by the human workers they replace.

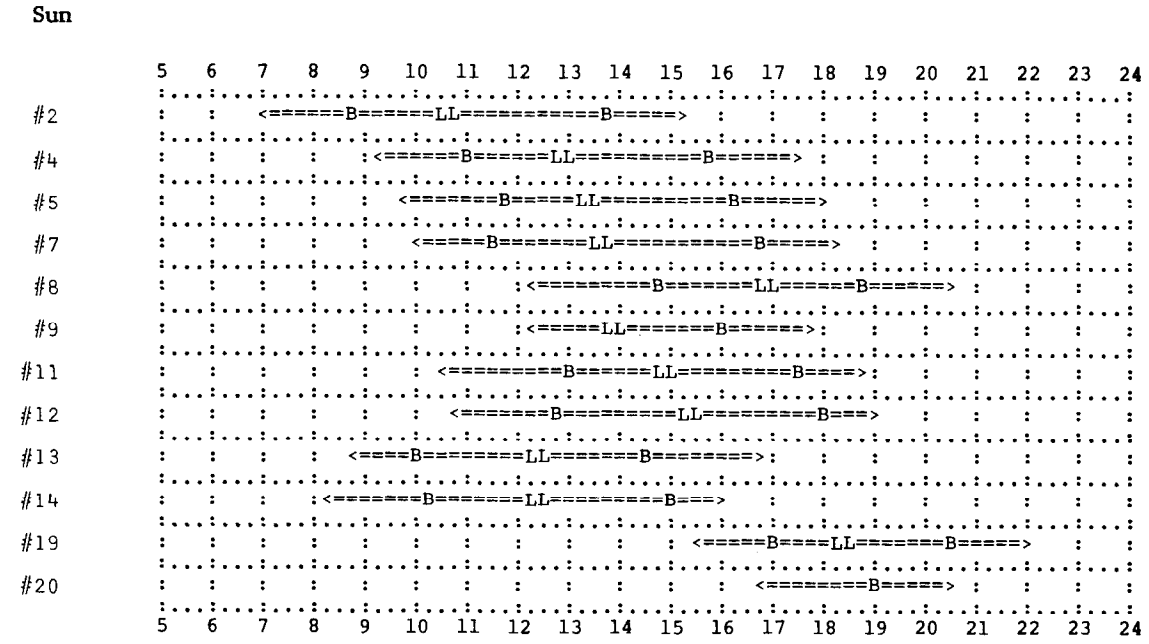
FIGURE 2

Scheduled as follows	Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sched	Avail
#1 Barrentos I	off all day	07:00-15:30	07:00-15:30	07:00-15:30	07:00-15:30	07:00-15:30	off all day	40	40
#2 Grace P	07:00-15:30	off all day	off all day	09:45-18:15	09:15-17:45	09:45-18:15	07:00-15:30	40	40
#3 Knight B	off all day	10:30-19:00	09:45-18:15	10:00-18:30	09:45-18:15	10:00-18:30	off all day	40	40
#4 Frank L	09:15-17:45	10:30-19:00	10:30-19:00	10:00-18:45	10:00-18:30	off all day	off all day	40	40
#5 Johnson E	09:45-18:15	off all day	off all day	10:30-19:00	10:15-18:45	10:30-19:00	09:00-17:30	40	40
#6 Busick J	off all day	11:00-19:30	12:00-20:30	11:15-19:45	off all day	off all day	09:30-18:00	32	32
#7 Hart S	10:00-18:30	11:15-19:45	off all day	11:30-20:00	off all day	11:15-19:45	10:00-18:30	40	40
#8 Timm P	12:15-20:45	off all day	off all day	12:15-20:45	10:30-19:00	15:30-23:15	10:00-18:30	39.25	40
#9 Roy W	12:15-18:00	off all day	off all day	09:00-16:45	13:45-22:00	12:00-19:15	10:15-18:45	35	40
#10 Peck M	off all day	09:00-13:15	off all day	off all day	11:30-20:00	off all day	12:00-20:00	19.75	24
#11 Saraceno D	10:30-19:00	11:30-20:00	10:45-19:15	off all day	off all day	15:00-21:00	08:45-16:15	36.5	40
#12 Olivas S	10:45-19:15	13:30-22:00	11:15-19:45	off all day	off all day	off all day	11:15-18:45	31	32
#13 Mahoney M	08:45-17:00	09:45-18:00	off all day	off all day	off all day	off all day	15:45-23:15	22.5	32
#14 Metzenbaum A	08:15-16:15	13:30-20:00	off all day	off all day	off all day	off all day	09:15-16:15	20.5	24
#15 Ruybal P	off all day	15:30-23:15	12:30-21:00	15:15-21:00	off all day	off all day	off all day	20	40
#16 Marquez Y	off all day	off all day	10:00-18:15	off all day	08:15-16:15	16:45-20:45	off all day	19.25	20
#17 Brown T	off all day	14:30-21:00	off all day	off all day	15:30-23:15	off all day	10:45-17:00	19	40
#18 Drieling A	off all day	off all day	11:30-19:15	off all day	off all day	08:30-14:15	16:30-20:45	16.75	40
#19 Rivera B	15:30-22:15	off all day	off all day	15:30-23:15	off all day	off all day	07:15-11:15	17.5	40
#20 McEnany K	16:45-20:45	off all day	off all day	08:15-12:30	off all day	10:30-16:15	11:15-15:15	17.5	40
#21 Hager L	off all day	off all day	15:30-23:15	off all day	16:45-20:45	off all day	13:00-18:45	16.5	40
#22 Farina B	off all day	off all day	16:15-22:00	off all day	15:15-21:15	off all day	15:15-21:30	16.5	40
#23 Martinez M	off all day	off all day	07:00-11:30	off all day	off all day	09:15-17:30	16:15-20:15	16.25	40
#24 Meyer S	off all day	off all day	off all day	16:30-22:15	off all day	13:30-20:00	08:30-13:15	16	40
#25 Evasnik K	off all day	off all day	off all day	off all day	11:30-20:00	off all day	11:15-19:45	16	40

FIGURE 3

Sun		—Available—			start	quit	1st bk	lunch	2nd bk	hrs	(for wk)
#2	Grace P	07:00	16:00	40	07:00	15:30	08:45	10:30	13:45	8	40
#4	Frank L	00:00	24:00	40	09:15	17:45	11:00	12:45	15:45	8	40
#5	Johnson E	08:30	22:00	40	09:45	18:15	11:45	13:15	16:15	8	40
#7	Hart S	00:00	24:00	40	10:00	18:30	11:30	13:30	16:45	8	40
#8	Timm P	12:00	22:00	40	12:15	20:45	14:45	16:45	18:45	8	39.25
#9	Roy W	11:30	21:30	40	12:15	18:00		13:45	16:00	5.25	35
#11	Saraceno D	08:15	21:15	40	10:30	19:00	13:00	14:45	17:30	8	36.5
#12	Olivas S	00:00	24:00	32	10:45	19:15	12:45	15:15	18:00	8.	31
#13	Mahoney M	07:15	19:15	32	08:45	17:00	10:00	12:15	14:30	7.75	22.5
#14	Metzenbaum A	00:00	24:00	24	08:15	16:15	10:15	12:15	15:00	7.5	20.5
#19	Rivera B	08:00	23:00	40	15:30	22:15	17:00	18:15	20:30	6.25	17.5
#20	McEnany K	11:00	23:45	40	16:45	20:45	19:00			4	17.5

FIGURE 4



14:45

Just as industrial robots replace some production line workers, so also semi-intelligent machines are replacing some managers. As they perform their management functions these machines don't look like managers and we therefore don't think of them as managerial robots—front office counterparts to their production line cousins.

But appearance has nothing to do with function in this domain. "Mailmobiles" which replace delivery boys at the Pentagon and at Citibank [6] don't look like traditional robots, but they are routinely referred to as robots. Similarly, super high level computer language compilers that produce software are sometimes referred to as robot programmers—machines that don't look at all like production line robots.

We suggest that the analogy between industrial robots and managerial robots should be recognized, and the "creatures" themselves should be given a name.

Managers perform a number of functions (few of which seem threatened by machines) such as organizing, strategic planning, motivating and developing human resources—the leadership role.

However, operations level planning, like strategic planning, is a management function, and semi-intelligent machines that produce operational level plans which human managers would otherwise produce can have the essential attributes of managerial robots.

Just as an industrial robot might replace a production line worker in the performance of a task or series of tasks, so also the automatic scheduler described above replaces an operative level manager in the performance of a specific management planning function. Our system does permit selected employees to be scheduled "manually" by the user, seated at the CRT terminal. This allows the human manager always to be in control if he wishes, to entertain "what if" possibilities, to introduce special overrides for special situations and the like.

There are, of course, other systems that do automated management planning. And for systems that fall into that class the name managerial robot seems appropriate.

We suggest that to qualify as a managerial robot a machine should have the following attributes:

1) It must perform a management planning function which would otherwise be performed by a human manager;
2) It must replace a manager in so far as that function is concerned, not simply supplement that manager;
3) It must exhibit a competence that rivals or surpasses that of a human performing the same function. It must not merely plan as fast or faster, it must plan as well or better;
4) It must exhibit the ability to solve new planning problems without modifications to the software. That is, within a range of planning settings the managerial robot must be able to produce a plan, a solution, without reprogramming;
5) It must handle problems that make significant demands on the intellectual energy and ability of a human planner or staff of planners. One may "plan" whether to take the stairway or the elevator, but this is not the type of high level, intellectually demanding task to which we make reference. Similarly, it may take mental energy to add a long column of figures, but this does not utilize what we would call "intellectual" ability.

We suggest one final criterion, not as an essential requirement, but as a highly desirable attribute of any planner, human or robot; it should provide prescriptions that are valuable,

that can make a significant difference in the effectiveness or efficiency of an organization that heeds these prescriptions. Our employee scheduling planner, described above, satisfies this criterion.

We feel that automatic bill-of-materials processors do not qualify as managerial robots for the reason that performing the bill-of-materials explosion is a clerical, rather than a management function. MRP I, material requirements planning, and MRP II, manufacturing resource planning come closer, but still at their present day level of operations do not qualify. Their outputs are not full fledged plans but more in the nature of augmented transaction records (and simple projections). Before reaching the stage of an executable plan, these records must be transformed into a coordinated master production schedule. In the few cases where this transformation is being achieved by "intelligent" computer systems, the term managerial robot would apply.

We suggest that, like managers themselves, managerial robots must be able to process new inputs to produce new outputs, without reprogramming. In addition, just as industrial robots are "general problem solvers" within a range of functions, so also managerial robots should be general problem solvers, able to accommodate to new planning problems through changed parameter settings. A machine lacking this capability would fail to measure up to the level of intelligence required of the managerial robot we envision.

MANAGERIAL ROBOTS VERSUS DECISION SUPPORT SYSTEMS

By the first criterion above, systems that solve problems are not managerial robots unless in solving those problems they do what we would call "managing" if done by a human.

By the second criterion, machines that only supplement or support management decision making would not qualify. Thus, planning systems like IFPS, the interactive financial planning system of Execucom, are not managerial robots. IFPS is a decision support system (DSS), and decision support systems do not replace managers. The automated components in decision support systems replace clerks who search files, perform analyses and prepare graphics in support of managerial decision making. We might find it useful to coin a phrase like "clerical robots" for automated spread sheets, database management systems, and modelling systems like IFPS.

Students of decision support systems focus on the dynamics of the decision making process, and on that part of decision situations that is judgmental rather than structured. They feel that the structured part may lend itself to automation but that the judgmental part should be left to the manager [7].

We support that position but we suggest that the manager to whom the judgmental part should be left, may be a robot manager.

Some will argue that if the judgmental component of the decision setting could be left to a robot manager, then it was not judgmental at all. But the continual redefinition of "judgmental" in an effort to put it just beyond the reach of the machine will get us nowhere. We suggest that managerial robots are invoking judgment any time they replace managers who invoke judgment. We don't redefine craft skill each time an industrial robot replaces a craftsman.

Decision support systems are designed to improve effectiveness in decision making in settings which are more unstructured, unstable and cannot be easily predefined. Managerial robots, for now at least, will prove more effective in rather structured, stable settings in which the requirements can be more nearly predefined. But it is in its replacing of the

manager that a managerial robot contrasts with decision support systems, which only support the manager as he or she interacts with the system.

MANAGERIAL ROBOTS AND "EXPERT SYSTEMS"

Nor do we consider managerial robots to be "expert systems." The term "expert systems" grew out of efforts by Simon and Newall to determine the nature of "expert" thought in physics and chess playing. Researchers in artificial intelligence now employ the term to designate intelligent machines that serve as aids in human problem solving. Examples include products of the "heuristic programming project" of the Department of Computer Science at Stanford University such as MYCIN—a system that performs consultations with a physician about infectious diseases [9]. The power of such systems is primarily dependent on the quantity and quality of their "knowledge bases," including their inference procedures. Knowledge is viewed as consisting of facts on the one hand, and heuristics on the other. Buchanan and Feigenbaum state that the facts constitute a body of knowledge about the task the system is to perform—facts that are generally agreed upon by experts in the field. The heuristics are rules of good judgment that characterize expert level decision making in a field [1].

To qualify as an expert system the system must be able to suggest promising ideas to the user—to draw inferences; and the system must be able to acquire new knowledge, acting autonomously, as users interact with it.

A decision support system contains a knowledge base—facts about entities and relationships between them which are relevant to the decision making they support. But they lack the inference procedures which characterize expert systems. Inferring promising avenues is left to the users. And by what we perceive to be the contemporary definition of DSS, to qualify a system does not require a mechanism for the autonomous acquisition of new knowledge through use.

Expert systems, like decision support systems, support but do not replace problem solvers. In this respect both are distinguished from managerial robots.

Managerial robots are more akin to expert systems than to decision support systems in that they do indeed draw inferences. Consider again our automatic scheduler.

As the construction of a schedule goes forward, our automatic scheduler considers the requirements which are as yet unmet, the remaining availabilities of the employees, and the union/management rules governing schedule design, and infers both the proper dimensions of the next shift and the employee it should be assigned in much the same way that an automated chess player selects its next move.

Our schedule does not presently augment its knowledge base through use, but it could be endowed with that capability. The quality of its schedules and the time required to produce them might be improved by saving past schedules and the requirements and employee availabilities which were processed to produce them. Then by analyzing the current pattern of requirements and the current pattern of employee availabilities it might select a previously produced schedule with comparable requirements and employee avail-