



## Jobs and tool sequencing in an automated manufacturing environment

P. CHANDRA, S. LI and M. STAN

A practical approach is presented for determining the sequence of jobs and tools in the magazine that would be required to process the jobs in an automated manufacturing environment. Each job has to be completed before a given due date. The magazine has a limited capacity necessitating setups which increase the lead times. Processing jobs also requires an appropriate fixture. Setting up a fixture also contributes to the setup times. A heuristic procedure is developed which determines the above sequences while minimizing the total setup and processing times. The performance of the heuristic is checked against optimal solutions for small-size problems while bounds are obtained (based on statistical lower bounding procedures) on the optimal solution for large-size problems. Computational results are provided for 155 test problems.

### 1. Introduction

Issues related to tool management are often ignored while making design and control related decisions on shop floors. By integrating decisions related to tool inventory, sequencing, monitoring, etc., with those concerning machines and parts scheduling, firms can reduce lead times and corresponding costs. Gray *et al.* (1992) present a taxonomy of such decisions in an automated manufacturing environment. In this paper we address the issue of simultaneous determination of job and tool sequences in an automated manufacturing facility. The scenario under consideration in this paper can occur in several industrial settings. Specifically, we have studied a particular configuration at a local plant which machines wings and other jobs for different aerospace firms. Each job arrives at this plant with a due date and requires one or more operations on one or more machines for completion. Each operation requires a set of cutting tools. Each machine has a local tool magazine with a capacity ranging from five to 30 tools. There also exists a central tool storage area on the shop floor which replenishes the local tool magazines on the machines. Both automatic and manual tool change systems exist on different machines. Jobs do not carry their own fixtures from one machine to another; instead, fixtures are located next to each machine. We address the problem of sequencing tools in the local tool magazine and jobs that need machining, simultaneously, so as to minimize the total time to set up the jobs on the fixture, load the tools on to the machine from the magazine and replace tools from the central tool storage area to the local magazine. Tool management issues at a machine level have started to attract more attention in recent years. Operations managers and researchers have now recognized that tool management affects the productivity of facilities, especially in automated manufacturing environments (Kiran and Krason 1988, Rhodes 1988, Martin 1989 and Gruver and Senniger 1990). The current research on tool management at the machine level focuses on four issues: (1) selection of tool equipment (e.g. tool magazines and automatic tool changers); (2) selection and placement of tools in a magazine; (3) tool replacement; and (4) tool sequencing on a flexible machine (Gray *et al.* 1992).

This research is related to tool sequencing. Research in tool and job sequencing on a flexible machine is quite recent and relatively sparse. Tang and Denardo (1988a, b) examined a single-machine problem with limited-size (i.e. capacity) tool magazine. They assumed that there exists a set of jobs each of which requires a specific set of tools for processing. The objective is to sequence the jobs and tools with a minimum number of tool switches. Bard and Feo (1989) studied a related problem of minimizing the total time for setups, tool replacement and machining for individual batches subject to a constraint on the capacity of the tool magazine. Their optimization algorithm requires the enumeration of all feasible solutions. Silver (1990) considered a problem of sequencing a family of parts on a single machine where the production rate for each part is taken as a control variable. Potential cost savings are identified by slowing down production rate of a key part in the family. Mittal and Lewis (1989) developed a mixed-integer programming model to minimize the total machine time, the tool change time, and the tool travel time. The problem is constrained by the tool-life economics and tool changes due to wear. De Werra and Widmer (1990) provided models for



a tool-loading problem in a FMS where tools can be replaced only at the end of a given period. Different objectives are discussed but no computational results are provided. Widmer (1991) used tabu search for solving scheduling problems with tool restrictions in a FMS. Veeramani (1992) examined the need for tool rationalization and describes the steps that are needed for integrating it in a concurrent engineering environment. He provides a new graph-theoretic method for clustering tools in order to facilitate the standardization of tool inventory. For a comprehensive survey of the tool management issues, the reader is also referred to Veeramani *et al.* (1992).

In this paper a problem is considered that is similar to Tang and Denardo (1988a, b) but incorporates two additional restrictions. First, each job to be processed may require a different fixture. For example, in machining operations, jobs of different shapes cannot be processed unless specialized fixtures are used to hold them. Second, each job has a given due date. It is interesting to note that few papers on tool sequencing incorporate due dates. It is known that the tool sequencing problem is an NP complete problem. The addition of due dates makes this problem extremely difficult to solve.

In the next section the proposed model is described in detail. In § 3 two different solution procedures for solving the job and tool sequencing problem are outlined. Section 3.1 describes a dynamic programming based procedure for solving small-size problems while a randomization heuristic, for solving large-size problems is given in § 3.2. An example is provided in § 4 and computational results are discussed in § 5.

## 2. Problem description

In this section, a single-machine job and tool sequencing problem is described. Consider a set of jobs each of which requires one operation on the machine and which must be processed before its due date. In order to perform the operation on each job, the machine requires a specific tool for processing and a specialized fixture to hold the job. Different jobs may require the same tool or the same fixture. This assumption is reasonable and intuitive since operations on some jobs or the job geometry may be similar. Also, different jobs may have the same due date. For example, different jobs may form subcomponents of an assembly. The magazine in the machine has a limited capacity to store tools. In this paper, it is assumed that each tool requires the same amount of space in the magazine so that its capacity can be defined by the maximum number of tools that the magazine can hold. Hence, the tool change time has two components: (1) time required to replace the tool on the machine from the magazine if the next job requires a different tool from the job that is being processed currently; and (2) time required to bring a tool from the tool room (i.e. the central tool storage area) if the tool required for the next operation is not in the magazine. Intuitively, the tool change time can be reduced by sequencing those jobs that require the same tools adjacent to each other if the required fixtures and the due dates are not considered.

The problem becomes complicated when the fixture change time is included. As mentioned earlier, different jobs may need different specialized fixtures. Thus, a certain amount of time is wasted in changing fixtures. Similarly, if only the fixture change time is considered, we can reduce it by sequencing the jobs that require the same fixture adjacent to each other.

The issue is made more complex when the tool change time, the fixture change time and the due dates are considered simultaneously. The objective of the problem is to find an optimal sequence of jobs that minimizes the total time of changing tools and fixtures and guarantees that jobs will be processed before their due dates.

Next, we introduce the notation and provide a nonlinear programming model for this problem.

### *Parameters*

$N$  = number of different jobs that need to be processed

$M$  = number of different tools that are required to process the  $N$  jobs

$L$  = number of different types of fixtures that are required to hold the  $N$  jobs  
 $C$  = maximum number of tools that the local tool magazine can hold

$i, j$  = index for jobs,  $i=1, \dots, N$ ;  $j=1, \dots, N$



$r$ =index for tools,  $r=1, \dots, M$

$s$ =index for fixtures,  $s=1, \dots, L$

$d_i$ =due date of job  $i$

$P_i$ =processing time of job  $i$

$S$ =time required to bring a tool from the tool room to the tool magazine  $U$ =time required to replace the tool on the machine from the magazine  $V$  = time required to change a fixture on the machine

$$b_{ir} = \begin{cases} 1 & \text{if job } i \text{ requires tool } r \\ 0 & \text{otherwise} \end{cases}$$

$$f_{is} = \begin{cases} 1 & \text{if job } i \text{ requires fixture } s \\ 0 & \text{otherwise} \end{cases}$$

$e$  =  $1 \times M$  matrix with element 1

$H$  = a very large positive number

*Variables*

$$Z_{ji} = \begin{cases} 1 & \text{if job } j \text{ precedes job } i \\ 0 & \text{otherwise} \end{cases}$$

$$Y = \begin{cases} 1 & \text{if a tool is brought from the tool room to process job } i \\ 0 & \text{otherwise} \end{cases}$$

$$T_i = \begin{cases} 1 & \text{if the current tool on the machine is replaced with another in the magazine to process job } i \\ 0 & \text{Otherwise} \end{cases}$$

$$F_i = \begin{cases} 1 & \text{if a fixture change is required when job } i \text{ is processed} \\ 0 & \text{otherwise} \end{cases}$$

$X_i$  = start time to process job  $i$

$TP_i$  = total processing and set up time for job  $i$

$M_i$  =  $M \times 1$  matrix indicating the status of the tool magazine (i.e. which tools are in the magazine) when job  $i$  is processed. The value of an element in the matrix is either 1 or 0 depending on whether the tool is in the magazine or not when job  $i$  is processed.

Hence, we can define:

$$W_{ji} = b_{ir} - \sum_{j \neq i} Z_{jr} b_{jk} = \begin{cases} 1 & \text{if } r \neq k \\ 0 & \text{otherwise} \end{cases}$$

$$Q_{jt} = f_{is} - \sum_{j \neq i} Z_{jt} f_{js} = \begin{cases} 1 & \text{if } s \neq t \\ 0 & \text{otherwise} \end{cases}$$

2.1. The model

The job and tool sequencing problem can be formulated as follows:

(PI)  $\min \sum_i (SY_i + UT_i + VFi)$

subject to

$$eM_i \leq C \quad i = 1, \dots, N \quad (1 \text{ a})$$

$$W_{ji} \leq T_i \quad i = 1, \dots, N; j = 1, \dots, N \quad (1 \text{ b})$$

$$Q_{ji} \leq F_i \quad i = 1, \dots, N; j = 1, \dots, N \quad (1 \text{ c})$$



$$e[M_i - \sum_{j \neq i} (Z_{ji} M_j)] + \leq CY_i \quad i = 1, \dots, N \quad (1 \text{ d})$$

$$SY_i + UT_i + VF_i + P_i = TP_i \quad i = 1, \dots, N \quad (2 \text{ a})$$

$$X_i + TP_i \leq d_i \quad i = 1, \dots, N \quad (2 \text{ b})$$

$$X_i - X_j \geq TP_j - H(1 - Z_{ji}) \quad i = 1, \dots, N; j = 1, \dots, N \quad (2 \text{ c})$$

$$X_i, TP_i \geq 0$$

$$W_{ji}, Q_{ji}, Z_{ji}, Y_i, T_i, F_i \in \{0, 1\} \quad (3)$$

The constraints fall into two categories: the first group (1 a-d) is concerned with the capacity of the magazine and the tool and fixture changes while the second group (i.e. 2 a-c) satisfies the due dates and establishes the precedence relationships.

Constraint (1 a) is the magazine capacity constraint where  $M_i$  indicates the tools in the magazine when job  $i$  is processed. The product of  $eM_i$  represents the total number of tools in the magazine. This number cannot exceed the magazine capacity,  $c$ . In constraints (1 b) and (1 c), respectively, the integer variable  $T_i$  or  $F_i$  is forced to be 1 if a tool switch or a fixture change is required else it is set to 0. 1' in constraint (1 d) becomes 1 if the tool needed to process job  $i$  is to be brought from the tool room and is 0 otherwise. Note that  $M_j$  indicates the tools in the magazine when job  $j$  is processed and  $M_i$  indicates the tools in the magazine when job  $i$  is processed. Constraint (1 d) also ensures that the number of tools that can be replaced in the magazine at a time does not exceed the magazine capacity,  $c$ . In the second group, constraint (2 a) sums up the total time required to process job  $i$ . Constraint (2 b) ensures that job  $i$  is completed before its due date while constraint (2 c) enforces the precedence relationship. The non-negativity requirements are given in constraint (3).

The objective function minimizes the total time for tool replacement from the central tool storage, tool change on the machine and fixture change. Notice that **PI** is a nonlinear integer problem. There exist computational difficulties in solving **PI** directly given that a number of variables are implicit (e.g.  $M_i$  and  $M_j$ ;  $Q_{ji}$  and  $W_{ji}$ ). However, when the number of jobs is small (say,  $< 10$ ), we can solve **PI** using dynamic programming, which is discussed in § 3. Large-size problems are solved using heuristics that are also described in the next section.

### 3. Solution procedures

In this section, two solution procedures that are developed to solve different size problems are discussed. For small-size problems, a dynamic programming procedure is used to obtain optimal solutions. For large-size problems, a 'randomization heuristic' has been developed to obtain bounds on the optimal solution. In § 3.1, the dynamic programming procedure is discussed and in § 3.2 the 'randomization heuristic' is presented.

#### 3.1. Dynamic programming procedure

The following additional notation is introduced:

- $C_i$ : completion time of job  $i$ ,  $i = 1, 2, \dots, N$
- $\{1, 2, 3, \dots, N\}$ : set of  $N$  jobs to be processed
- $(1, 2, 3, \dots, N)$ : optimal sequence for jobs  $1, 2, 3, \dots, N$
- $Q$  = set of jobs containing job  $i$

$\lceil(Q)$  = minimum time sequence which starts with a dummy job  $O$  and ends at job  $i$ .

Thus for any set  $Q$ , the objective is to choose a job  $i$  that will minimize the total tool and fixture change time for the set  $Q$ . The total time is the sum of processing and set up time required by job  $i$  plus the minimum time of an optimal sequence of jobs in  $Q$  without job  $i$ .

The recursive expression is given as follows:

$$\lceil(Q) = \min_{\substack{i \in Q \\ C_i \leq d_i}} \{ \lceil(Q - \{i\}) + TP_i \}$$



Notice that  $\lceil (Q - \{i\}) \rceil$  is the minimum time found by optimally sequencing jobs in set  $Q$  (without job  $i$ ). An implied restriction is that for a sequence of  $(1, 2, 3, \dots, N)$ , due dates of all the jobs should be satisfied. It is easy to verify that the dynamic programming approach can only be used to solve small-size problems since the complexity of a problem with  $N$  jobs is  $O(N^2 2^N)$ . However, the optimal solutions obtained from the dynamic programming solution procedures can be used to examine the quality of our heuristic solutions for small-size problems. The results are discussed in § 5.

### 3.2. Randomization heuristic

In this section an approximate procedure for solving the job and tool sequencing problem is outlined. Statistical lower bounds are determined, based on the ordered sequence of the solutions obtained from the randomization process. The randomization heuristic takes the due dates, time to process the jobs and their tool and fixture requirements as given. It simultaneously determines the sequence of jobs for processing and the sequence of tools that are needed for processing so as to minimize the completion time (total setup times for changing tools and fixtures).

A 'period' is defined as the time slot between two consecutive due dates. Let  $t = \text{index}$  for a period which ends on the  $t$ th due date, i.e.

$t = 1$  represents the time interval between zero and the first due date

$t = 2$  represents the interval between the first and second due date; etc. The randomization heuristic can be described as follows:

*Step 1.* Find an initial feasible solution

- .Schedule jobs according to the earliest due date rule.
- Perform local improvement: find the optimal sequence within each period by switching jobs if it leads to reduction in the completion time. We define  $T_{\text{best}}$  and  $Z_{\text{best}}$  to be the best solution obtained by our heuristic and the corresponding objective function value respectively. These are initialized as follows:

$T_{\text{best}}$  = best initial feasible schedule

$Z_{\text{best}}$  = objective function value (i.e., completion time) for  $T_{\text{best}}$

Also, let  $Z_{\text{nextbest}} = \infty$  and the iteration counter,  $iter$ , be set to 1.

*Step 2.* Improvement of the initial solution by a semi-greedy heuristic.

- Initialize the iteration parameters
  - Let  $iter$  and  $Z_{iter}$  be the selected schedule and the corresponding objective function value found in the previous iteration of step 2. When  $iter = 1$ ,  $Z_{iter} = Z_{\text{best}}$  and  $T_{iter} = T_{\text{best}}$
- Generate new solutions
  - For every  $t$  and  $t'$  where  $t$  and  $t'$  are two periods such that  $t < t'$ , insert every job  $j$  currently scheduled in  $t'$  before every job  $i$  currently scheduled in  $t$ .
  - Determine the slack time (due date-completion time) for the last job currently scheduled in period  $t$ .
  - If the (slack time  $<$  process time for job  $j$ ) then the insertion is 'definitely' infeasible else check for feasibility (which is described later).
  - Store feasible insertions and the accompanying objective function values.
- Partition the new feasible solutions (obtained via the insertion process as shown above) into two sets,  $S_1$  and  $S_2$  where

$S_1$  = set of all new feasible solutions whose objective function value is  $< Z_{iter}$ .  $S_2$  = set of all new feasible solutions whose objective function value is  $\geq Z_{iter}$ .

Also, let  $Z_{\text{lowest}}$  = the solution with the lowest objective function value found in the current iteration (i.e.  $Z_{\text{lowest}} \in \{S_1 \cup S_2\}$ ).



- Neighborhood selection process
  - If  $S1 \neq \Phi$  then create a 'neighborhood' that contains all the feasible new solutions (belonging to  $S1$ ) whose objective function value is within  $P\%$  of  $Z_{lowest}$ .
  - If  $S1 = \Phi$  then the neighborhood contains the new feasible solutions (belonging to  $S2$ ) which are within  $P\%$  of  $Z_{lowest}$  (i.e.  $> Z_{lowest}$  by at most  $p\%$ ).
- Update solution .
  - Increase the iteration count,  $iter$ , by 1.
  - Choose one solution randomly from the selected neighborhood. Let  $T_{iter}$  = new randomly selected solution  
 $Z_{iter}$  = new objective function value for the selected solution .
  - If ( $Z_{lowest} < Z_{best}$ ) then  $Z_{next\ best} = Z_{best}$ ; and  $Z_{best} = Z_{lowest}$   
If ( $Z_{lowest} > Z_{best}$  and  $Z_{lowest} < Z_{nextbest}$  then  $Z_{nextbest} = Z_{lowest}$
- Check for the prescribed stopping limit (number of iterations).  
If the iteration count,  $iter <$  prescribed stopping limit, repeat step 2, else go to step 3.

Step 3. Calculate lower bounds using  $Z_{best}$  and  $Z_{nextbest}$

If  $LB = UB$ , stop else repeat step 2 for a prescribed additional number of iterations and stop.

The randomization heuristic tries to overcome the disadvantages of a greedy heuristic that often pushes the solution in a corner from which it is unable to get out. Since the starting solution for the next iteration is not necessarily the lowest solution of the last iteration, the heuristic will quite often take the solution process out of a local minima.

### 3.3. Lower bounding procedure

The heuristic provides us with an upper bound on the optimal solution value of the jobs and tool sequencing problem. A deterministic data dependent lower bound is extremely difficult to obtain. In order to get some sense of the solution quality and to measure how close the heuristic solution is to the optimal value, statistical methods were used to obtain lower confidence limits at a given significance level.

For any given instance of the problem, let  $z$  be the value of the objective function obtained by the randomization heuristic (i.e.  $Z_{best}$  once the procedure is terminated). Let  $z^*$  be the lower confidence limit on the unknown optimal solution  $z^*$ . The aim is to calculate the value of  $z^*$  such that

$$\Pr(z^* \geq \check{z}^*) \geq (1 - \alpha)$$

$\check{z}^*$  is the statistical lower bound at significance level  $1 - \alpha$ . Each update of  $Z_{best}$  at any iteration of the randomization heuristic is added to a set of solutions obtained via the local improvement process. Let there be  $n$  such solutions when the procedure terminates. Following Derigs' (1985) notation, let the  $n$  solutions be  $S_1, S_2, \dots, S_n$  and their respective objective function values (i.e. completion times) be  $z_1, z_2, \dots, z_n$ .  $S_{(1)}, S_{(2)}, \dots, S_{(n)}$  and  $Z_{(1)}, Z_{(2)}, \dots, Z_{(n)}$  denote the ordered sequence such that  $Z_{(i)} \leq Z_{(i+1)}$  for  $1 \leq i \leq n - 1$ . Using  $n$  samples we construct a  $100(1 - \alpha)\%$  lower confidence limit,  $\check{z}^*$ , on the unknown optimal solution to the overall sequencing problem,  $z^*$ . The confidence level  $\alpha$  may be interpreted as meaning that  $\check{z}^*$  will fail to be the correct lower bound only  $100\alpha\%$  of the time.

Several different confidence limit procedures have been developed in literature (see Monroe 1982). We employ the 'limiting distribution approach' to calculate  $\check{z}^*$  whereby

$$\check{z}^* = z_{(1)} - C_\alpha(z_{(k)} - z_{(1)})$$

Such that

$$\Pr(z_{(1)} - C_\alpha(z_{(k)} - z_{(1)}) \leq z^*) = 1 - \alpha$$

This approach is based on order statistic rather than on a distribution assumption of the solution domain. For purposes of illustration, two procedures based on the 'limiting distribution approach' are employed for obtaining  $\check{z}^*$ . Robson and Whitlock (1964) and Boender *et al.* (1982), have given the following approaches for calculating the  $100(1 - \alpha)\%$  lower confidence limit for  $z^*$  that differ in the choice of values for  $k$  and  $C_\alpha$  :



Robson and Whitlock (LB1)

(i)  $k=2$ ; and (ii)  $C_\infty=(1-\infty)/\infty$   
Boender et al. (LB2)

(i)  $k=2$ ; and (ii)  $C_\infty = (1/((1-\infty)^2-1))$

Both the above formulae are based on the 'best' and the 'next best' upper bound value obtained from the randomization procedure. Consequently, our heuristic stores only these two variables and updates them at each iteration. Finally, upon termination of the procedure, the lower bound estimate,  $\check{z}^*$ , is calculated and compared with the upper bound obtained through the heuristic. In § 5 we explain the application of these procedures for obtaining statistical lower bounds.

3.4. Check for feasibility and complexity

At any instant, the state of the system is characterized by the following parameters: the fixture, the tools in the magazine, and the tool in the machine. Each time a job is processed the state of the system is updated in terms of the fixture on the machine, the tool in the machine and the tool mix in the magazine.

Every time two consecutive jobs require different tools and the tool needed for the incumbent job is not in the magazine, we face an optimization problem: which tools to remove from the magazine and which tools to bring from the tool room in order to minimize the number of tool changes in future (and consequently the objective function value). We remove the tools which are least needed and bring those which are most needed. To obtain the number of occurrences for each tool, we scan the solution from the point when a needed tool is not found in the magazine until the last job and determine the future need for this tool. These computations are performed with a worst case complexity of  $O(N)$ .

Once a new solution is obtained, we verify whether each of the characteristics of the state of the system are compatible with the job being inserted. If anyone of these requirements for processing (i.e. the availability of the required tool in the machine/magazine or the fixture on the machine) is not satisfied then we change the state by bringing the tool to the magazine and/or exchanging the fixture and/or the tool. These modifications may increase the completion time of jobs in the period in which a new job is inserted and hence the lateness of one of the jobs in this period may become positive. This is why the slack time (due date of last job in a period -completion time of the last job in that period) gives only the necessary condition for feasibility.

As the jobs are checked one at a time, the overall complexity of checking the feasibility and computing the objective function value of a solution is no more than  $O(N^2)$  (or  $O(N)$  if the exchange of tools is not required). The complexity of the algorithm also depends on the number of solutions considered. This number is given by

$$\left( \sum_{\substack{t,t' \\ t < t'}} (R_t R_{t'}) < N^2 \right)$$

where  $R_t$  and  $R_{t'}$  are the cardinality of the set of jobs currently scheduled in periods  $t$  and  $t'$  respectively. This gives an overall complexity of  $O(N^4)$  for our heuristic.

4. A numerical example

In this section, an example is provided to show the implementation of the heuristics developed in this paper. The data of this example are as follows:

- Number of jobs: 8
- Number of tools: 5
- Number of fixtures: 4



Capacity of the magazine: 3  
 Initial tool status of the magazine: (1, 2, 5)  
 Initial fixture on the machine: 3  
 Initial tool in the machine: 1

For every problem a dummy job is created (whose processing time and due date are set to zero) to initialize the tool and fixture status. In this example, it is assumed that at the beginning the tools in the magazine are 1, 2, and 5; the tool and fixture on the machine are 1 and 3, respectively. This means that the dummy job requires tool and fixture 3. Table 1 provides the tool and fixture requirements and the processing time for this example. Note that the first job is the dummy job. The three different due dates represent the three periods. The first period ( $t = 1$ ) contains the dummy job with due date of zero. The second period ( $t = 2$ ) contains four jobs (2-5) with a due date of 60. The third period ( $t = 3$ ) has four jobs (6-9) with a due date of 112. In this example, the tool change time in the machine is 4; tool replacement time to the tool room is 8 and fixture change time is 5. Following the EDD rule and local improvement, the initial solution is obtained as follows:

Initial solution:  $T_{best} = (1, 3, 2, 5, 4, 9, 7, 6, 8)$   
 Objective function value of the initial solution:  $Z_{best} = 101$

	Job								
	1	2	3	4	5	6	7	8	9
Tool requirement	1	1	4	5	6	7	8	9	5
Fixture requirement	3	2	4	4	2	1	4	1	3
Processing time	0	2	2	6	3	5	4	7	2
Due date	0	60	60	60	60	112	112	112	112
Period ( $t$ )	1	2	2	2	2	3	3	3	3

Table 1. Data for the eight-job example.

The total time required to complete the eight jobs is 101. Note that, in this heuristic, every time the tool room is visited two tools are replaced at a time in the magazine. The two tools brought in are the ones which will be used most recently and frequently. Like any other replacement rule, this could sometimes lead to an inferior starting solution.

Next, efforts are made to improve the initial solution by looking to insert jobs from each period ( $t' > 2$ ) before another job in an earlier period ( $t > 1$ ).  $T_{iter} = T_{best}$  and  $Z_{iter} = Z_{best}$  for  $iter = 1$ . Since it might not be feasible to insert a job from a later period to an earlier, the feasibility is examined. This is done in two stages. First, calculation of slack time provides a way of evaluating the necessary condition for any feasible insertion. The slack time is the difference between the due date and the completion time of the last job in that period. If the slack time is greater than the processing time of the job to be inserted then the job could be possibly shifted, and then the feasibility is checked by considering the tool and fixture changes in all subsequent periods.

Suppose, in the current iteration, we want to insert job 9 from period 3 before job 3 in period 2. Hence, slack time = the due date for job 4 (i.e. the last job currently scheduled in period 2) - the completion time for job 4. The completion time of job 4 is 52 (which can be obtained by adding the time spent in one tool room visit, four tool changes, three fixture changes and the total processing time of jobs 3, 2, 5, and 4). Since due date for period 2 is 60, the slack time is 8 (= 60 - 52) which is greater than the processing time of job 9 (i.e. 2). Hence, we conclude that the insertion of job 9 before job 3 may be feasible (the same is true for inserting job 9 before jobs 2, 4 or 5). Now the feasibility of this possible new solution (i.e., 1, 9, 3, 2, 5, 4, 7,



6, 8) is checked by determining the additional time required for tool and fixture changes and by checking if there is a violation of any due date. This sequence is found to be feasible so job 9 is inserted before job 3 and this new solution generates an objective function value of 95 (setup time = 64 and processing time = 31). Since the new solution is better than  $Z_{iter}$  (i.e. 101) it is stored in the set  $S_1$  (which contains those new solutions from the current iteration that are found to be better than the current solution).

In this iteration (i.e.  $iter = 1$ ) two other solutions from other insertions whose objective function values are better than  $Z_{iter}$  are found. These values were 96 and 100. Both these solutions are added to the set  $S_1$  whose cardinality, now, becomes three. Next, a neighborhood of the current best solution in  $S_1$  (i.e.  $Z_{lowest} = 95$ ) is created which consists of all the solutions which are within 15% (our  $p\%$  value) of 95, i.e. any solution whose objective function value is  $\sim 109.25$ . In this example, all the three solutions in  $S_1$  fall in this neighborhood. One solution is chosen randomly for this neighborhood which forms the starting solution for the next iteration. In this example, the solution associated with the objective function value of 96 was chosen randomly. Consequently, for  $iter = 2$ ,  $Z_{iter} = 96$ . In this case,  $Z_{best}$  (which has a value of 101) is updated by the value of  $Z_{lowest}$  i.e. 95.  $Z_{bestnext}$  is updated correspondingly.

The randomization heuristic is repeated for a number of iterations and several new solutions are generated. The final solution obtained is (1,9,4,7,3,5,2,8,6) with an objective value (or  $Z_{best}$  of 70. The dynamic programming procedure also gave the same solution (with the same objective function value) as the one obtained above by the randomization heuristic.

### 5. Computational experiments

In this section the results of computational experiments performed to solve the jobs and tool sequencing problem and to evaluate the performance of the heuristic described in § 3 are reported.

#### 5.1. Characteristics of problem parameters

Since the task of deciding whether a problem is feasible is as hard as solving it, only feasible test problems were generated. Three types of test problems were defined: loose, medium and tight. The difference in the three types of problems is primarily the difficulty in improving the initial solution obtained from the EDD rule. The loose, medium, and tight type problems consist of up to 7, 5 and 3 jobs, respectively, within each due date. The due dates are generated as tightly as possible within each category.

The test problem sizes range from small to large. Problems with less than 10 jobs are considered as small-size problems which are solved by both the dynamic programming and the randomization heuristic procedure. The solutions obtained by the heuristic were compared with the optimal solutions obtained through the dynamic programming procedure. The number of jobs in the large-size problems, that were solved by the heuristic, are set at 10,20,50 and 60. The magazine capacity varies with the number of tools it can hold. Different capacities (3, 7 and 10 tools) of the magazine were considered in the test problems. Similarly, different number of available tools (5, 10 and 25) and available fixtures (4, 5 and 10) were used in the computational experiments.



5.2. Description of the data sets

For different test problems, the various setup and process times were generated as follows:

(1) The processing time is generated from  $N(2,8)$  which is a truncated normal distribution ranging from 2 to 8.

No. of jobs	No. of tools	Magazine capacity	No. of fixtures	Type of problem	LB1	LB2	UB	Dynamic programming solution
6	5	3	4	Medium	66	66	66*	66
					25	25	25*	25
					47	47	47*	47
					28	28	28*	28
					43	43	43*	43
7	5	3	4	Medium	47	47	47*	47
					35	35	35*	35
					51	51	51*	51
					38	38	38*	38
					46	46	46*	46
8	5	3	4	Medium	47	47	47*	47
					44	44	44*	44
					55	55	55*	55
					52	52	52	48
					66	66	66	49
9	5	3	4	Medium	59	59	59*	59
					39	39	39*	39
					63	63	63*	63
					56	56	56*	56
								49
					49	49	49*	

\* Denotes optimal solution obtained through the heuristics.

Table 2. Computational results from small-sized problem.

(2) The time to change a tool on the machine from the magazine is generated from  $U(1, 5)$  which is a discrete uniform distribution ranging from 1 to 5.

(3) The time to bring a tool from the tool room and place it in the magazine is generated from  $U(5, 10)$ .

(4) The time to change a fixture is generated from  $U(5, 15)$ .

(5) The value of  $p$ , the parameter used to define the neighborhood, was set to 15%.

5.3. Experimental results

The randomization heuristic has been implemented such that a dummy job is introduced (as the first job of any schedule) with processing times and due dates equal to zero. This allows each job to be the first job (in a real schedule) corresponding to the depot in a transportation problem. This dummy job can represent the initial step or the state of system from an earlier period.

A total of 155 problems has been tested using the data sets generated as above. Computations were performed on a SPARC station 2. Twenty small-size problems were solved by both the



dynamic programming and the heuristic procedures. These results are reported in Table 2. It should be noted that for 18 of 20 problems, the optimal solution was obtained by the randomization heuristic. Tables 3-5 report the results for the large-size problems. In each table the number of the test cases whose upper bounds were found to be equal to their lower bounds is provided. Note that the sum of those numbers account for 80% of the total test problems. The two statistical

No. of problems	No. of jobs	Magazine capacity	No. of tools	No. of fixtures	No. of cases LB = UB		Deviation from the best LB			Mean computing time(s)
					LB1	LB2	Mean	Min	Max	
5	10	3	5	4	5	5	0	0	0	0.58
5		3	5	4	5	5	0	0	0	6.1
5	20	7	10	5	4	4	0.012	0	0.06	5.6
5		3	5	4	4	4	0.058	0	0.29	118.64
5	50	7	10	5	5	5	0	0	0	119.86
5		16	25	10	3	3	0.028	0	0.094	155.18
5		3	5	4	4	4	0.024	0	0.12	361.16
5	60	7	10	5	2	2	0.085	0	0.24	332.44
5		16	25	10	5	5	0.029	0	0.078	338.4

'Mean deviation from the best LB' means the mean of the deviations of five test problems between the upper and the best lower bound

$$\left( = \frac{1}{5} \left[ \sum_{i=1}^5 (UB_i - \max(LB1_i, LB2_i)) \right] \right)$$

where  $UB_i$ ,  $LB1_i$ , and  $LB2_i$  are upperbound and the two lower bounds respectively for problem  $i$ .

'Min deviation' means the minimum deviation between the upper and the best lower bound among five test problems

$$\left( = \min_i [UB_i - \max(LB1_i, LB2_i)] \right).$$

'Max deviation' means the maximum deviation between the upper and the best lower bound among five test problems

$$\left( = \max_i [UB_i - \max(LB1_i, LB2_i)] \right).$$



Table 3. Computational results on tight-type problems.

No. of problems	No. of jobs	Magazine capacity	No. of tools	No. of fixtures	No. of cases LB = UB		Deviation from the best LB			Mean computing time(s)
					LB1	LB2	Mean	Min	Max	
5	10	3	5	4	4	4	0.048	0	0.24	0.46
5		3	5	4	5	5	0	0	0	4.28
5	20	7	10	5	3	3	0.121	0	0.35	5.2
5		3	5	4	5	5	0	0	0	134.2
5	50	7	10	5	4	4	0.042	0	0.21	131.5
5		16	25	10	5	5	0	0	0	154.84
5		3	5	4	3	3	0.063	0	0.282	341.26
5	60	7	10	5	4	4	0.028	0	0.14	279.54
5		16	25	10	4	4	0.014	0	0.069	301.14

'Mean deviation from the best LB' means the mean of the deviations of the five test problems between the upper and the best lower bound

$$\left( = \frac{1}{5} \left[ \sum_{i=1}^5 (UB_i - \max(LB1_i, LB2_i)) \right] \right)$$

where  $UB_i$ ,  $LB1_i$ , and  $LB2_i$  are upperbound and the two lower bounds respectively for problem  $i$ .

'Min deviation' means the minimum deviation between the upper and the best lower bound among five test problems

$$(\text{Min deviation}) = \min_i [UB_i - \max(LB1_i, LB2_i)]$$

'Max deviation' means the maximum deviation between the upper and the best lower bound among five test problems

$$(\text{Max deviation}) = \max_i [UB_i - \max(LB1_i, LB2_i)]$$

Table 4. Computational results on medium-type problems.

lower bounds, i.e. LB1 and LB2 are derived using the lower bounding procedure discussed in the earlier section. While the mean deviations vary among test problems, the maximum mean deviation is about 10% between the upper and lower bounds. It should be noted that the two lower bounding formulae that have been used for the purposes of illustration are sensitive to the uniqueness of optimal solution. They provide a relatively accurate picture of the gap between the two bounds when multiple optima exist, especially when the value of the solution is not very large. The problems that have been solved have multiple optima. In case there exists a unique optimal solution then several other procedures can be used for determining the statistical lower bounds, which are discussed in Derigs (1985), and include those based on 'extreme-value theory' (e.g. the 'Weibull approach' of Golden and Alt 1979) or the 'goodness-of-fit' approach (e.g. Liao *et al.* 1973). The computational time of the heuristic increases as the number of jobs increase in the test problems. For the test problems with 10 jobs, the computational time is < 1 s. When the number of jobs increases to 20, the computational time is about 4-5 s. The 60 jobs test problems require about 5 min to solve. As a comparison, the dynamic programming procedure takes around 2 h for solving a 10- job problem.

In conclusion, a rather difficult problem often encountered in industry, and which has implications on the competitiveness of a firm in terms of cost and delivery times is addressed. Results show the effectiveness of the randomization heuristic for solving large-problems. The heuristic solution is also benchmarked against the optimal



No. of problems	No. of jobs	Magazine capacity	No. of tools	No. of fixtures	No. of cases LB = UB		Deviation from the best LB			Mean computing time(s)
					LB1	LB2	Mean	Min	Max	
5	10	3	5	4	5	5	0	0	0	0.46
5		3	5	4	5	5	0	0	0	4.4
5	20	7	10	5	4	4	0.033	0	0.165	5.9
5		3	5	4	3	3	0.10	0	0.457	131.6
5	50	7	10	5	3	3	0.049	0	0.144	119.1
5		16	25	10	4	4	0.014	0	0.071	192.4
5		3	5	4	4	4	0.129	0	0.64	345.28
5	60	7	10	5	2	2	0.06	0	0.159	287.9
5		16	25	10	5	5	0	0	0	315.6

'Mean deviation from the best LB' means the mean of the deviations of five test problems between the upper and the best lower bound

$$\left( =\frac{1}{5} \left[ \sum_{i=1}^5 (UB_i - \max(LB1_i, LB2_i)) \right] \right)$$

where  $UB_i$ ,  $LB1_i$  and  $LB2_i$ ; are upperbound and the two lower bounds respectively for problem i).

'Min deviation' means the minimum deviation between the upper and the best lower bound among five test problems

$$\left( =\min_i [UB_i - \max(LB1_i, LB2_i)] \right)$$

'Max deviation' means the maximum deviation between the upper and the best lower bound among five test problems

$$\left( =\max_i [UB_i - \max(LB1_i, LB2_i)] \right)$$

Table 5. Computational results on loose-type problems.

solution for small size problems. We feel that tool management is a critical area since firms spend large sums of money on tool management (e.g. annual tool-related expenses at a local firm is around \$ 5 million). Hence, good practices could lead to considerable reduction in costs associated with tooling. The next phase of this research is to incorporate the above model in a larger tool inventory management framework where the tool availability is limited by its need at multiple workstations.

**Acknowledgments**

This research was partially supported by N8ERC Grants OGPOO42150 (P. C.) and OGPO121644 (8. L.). The authors wish to thank Luc Guenette from CAE Electronics for suggesting the problem and for numerous discussions.

**References**

BARD, J., and FEO, T. A., 1989, The cutting path and tool selection problem in computer aided process planning. *Journal of Manufacturing Systems*, 8, 17-26.

BOENDER, C. G.E., RINNOY KAN, A. H. G., STOUIGIE, L., and TIMMER, G. T., 1982, A stochastic method for global optimization. *Mathematical Programming*, 22, 125-140.

DERIGS, U., 1985, Using confidence limits for global optimum in combinatorial optimization. *Operations Research*, 33, 1024-1049.

DE WERRA, D., and WIDMER, M., 1990, Loading problems with tool management in flexible manufacturing



systems: A few integer programming models. *International Journal of Flexible Manufacturing Systems*, 3, 71-82.

GOLDEN, B. L., and ALT, F.B., 1979, Interval estimation of a global optimum for large combinatorial problems. *Naval Research Logistics Quarterly*, 26, 69-77.

GRAY, A. E., SEIDMANN, A., and STECKE, K. E., 1992, A synthesis of design models for tool management in automated manufacturing. Working Paper QM 92-02, Simon Graduate School of Business Administration, University of Rochester, *Management Science* (forthcoming).

GRUVER, A., and SENNINGER, M. T., 1990, Tool management in FMS. *Mechanical Engineering*, 112 (3), 40-44.

KIRAN, A. S., and KRASON, R. J., 1988, automating tooling in a flexible manufacturing system. *Industrial Engineering*, April, 52-57.

LIAU, T., HARTLEY, H. O., and SIELKEN, R. L., 1973, Confidence limits for the global optima. Themis Technical Report No.43, Institute of Statistics, Texas A&M University.

MARTIN, J. M., 1988, Managing tools makes the cell click. *Manufacturing Engineering*, 4, 59-62.

MITTAL, R. O., and LEWIS, R. L., 1989, A micro process planning system based on integer programming for prismatic parts produced on horizontal machining centers. *Annals of Operations Research*, 17, 273-290.

MONROE, II. M., 1982, Confidence limits for the global optimum. PhD thesis, Texas A&M University.

ROBSON, D., and WHITLOCK, J., 1964, Estimation of a truncation point. *Biometrika*, 51, 33-39.

RHODES, J. S., 1988, FMS tool management systems. *Flexible Manufacturing Systems*, edited by Thomas I. Drozda (Dearborn, MI: SME Publication), pp. 269-286.

SILVER, E., 1990, Deliberately slowing down output in a family production context. *International Journal of Production Research*, 28, 17-27.

TANG, C. S., and DENARDO, E. V., 1988a, Models arising from a flexible manufacturing machine, part I: Minimization of the number of tool switches. *Operations Research*, 36, 767-777.

TANG, C. S., and DENARDO, E. V., 1988b, Models arising from a flexible manufacturing machine, part II: Minimization of the number of switching instances. *Operations Research*, 36, 778-784.

VEERAMANI, D., 1992, Rationalization of cutting-tool requirements. Working paper, University of Wisconsin-Madison.

VEERAMANI, D., UPTON, D. M., and BARASH, M. M., 1992, Cutting-tool management in computer-integrated manufacturing. *International Journal of Flexible Manufacturing Systems*, 3 (4), 237-265.

WIDMER, M., 1991, Job shop scheduling with tooling constraints: a tabu search approach. *Journal of Operations Research Society*, 42, 75-82.

GOLDEN, B. L., and ALT, F. B., 1979, Interval estimation of a global optimum for large combinatorial problems. *Naval Research Logistics Quarterly*, 26, 69-77.

GRAY, A. E., SEIDMANN, A., and STECKE, K. E., 1992, A synthesis of design models for tool management in automated manufacturing. Working Paper QM 92-02, Simon Graduate School of Business Administration, University of Rochester, *Management Science* (forthcoming).

GRUVER, A., and SENNINGER, M. T., 1990, Tool management in FMS. *Mechanical Engineering*, 112 (3), 40-44.

KIRAN, A. S., and KRASON, R. J., 1988, Automating tooling in a flexible manufacturing system. *Industrial Engineering*, April, 52-57.

LIAU, T., HARTLEY, H. O., and SIELKEN, R. L., 1973, Confidence limits for the global optima. Themis Technical Report No.43, Institute of Statistics, Texas A&M University.

MARTIN, J. M., 1988, Managing tools makes the cell click. *Manufacturing Engineering*, 4, 59-62.



- MITTAL, R. O., and LEWIS, R. L., 1989, A micro process planning system based on integer programming for prismatic parts produced on horizontal machining centers. *Annals of Operations Research*, 17, 273-290.
- MONROE, II. M., 1982, Confidence limits for the global optimum. Ph.D thesis, Texas A&M University.
- ROBSON, D., and WHITLOCK, J., 1964, Estimation of a truncation point. *Biometrika*, 51, 33-39.
- RHODES, J. S., 1988, FMS tool management systems. *Flexible Manufacturing Systems*, edited by Thomas I. Drozda (Dearborn, MI: SME Publication), pp. 269-286. SILVER, E., 1990, Deliberately slowing down output in a family production context. *International Journal of Production Research*, 28, 17-27.
- TANG, C. S., and DENARDO, E. V., 1988a, Models arising from a flexible manufacturing machine, part I: Minimization of the number of tool switches. *Operations Research*, 36, 767-777.
- TANG, C. S., and DENARDO, E. V., 1988b, Models arising from a flexible manufacturing machine, part II: Minimization of the number of switching instances. *Operations Research*, 36, 778-784.
- VEERAMANI, D., 1992, Rationalization of cutting-tool requirements. Working paper, University of Wisconsin-Madison.
- VEERAMANI, D., UPTON, D. M., and BARASH, M. M., 1992, Cutting-tool management in computer-integrated manufacturing. *International Journal of Flexible Manufacturing Systems*, 3 (4), 237-265.
- WIDMER, M., 1991, Job shop scheduling with tooling constraints: a tabu search approach. *Journal of Operations Research Society*, 42, 75-82.
- .
- .