TECHNICAL RESEARCH REPORT

Selecting Equipment for Flexible Flow Shops

by Rohit Kumar, Jeffrey W. Herrmann

TR 2003-19

Selecting Equipment for Flexible Flow Shops

Rohit Kumar
Institute for Systems Research,
University of Maryland, College Park, MD 20742, USA

Jeffrey W. Herrmann
Department of Mechanical Engineering and Institute for Systems Research,
University of Maryland, College Park, MD 20742, USA

Abstract

Equipment selection is one of the challenges faced during manufacturing system design. Selecting the proper equipment is important to satisfying budget constraints, achieving required throughput, and reducing manufacturing cycle time and inventory. This paper formulates an equipment selection problem and presents two search algorithms used to find high-quality solutions. Queueing system models are used to calculate the manufacturing cycle time. The paper discusses the results of experiments conducted to evaluate the performance of the algorithms across a range of problem characteristics.

Keywords: Manufacturing system design, equipment selection, discrete optimization.

Please send correspondence to
Dr. Jeffrey W. Herrmann
Department of Mechanical Engineering
University of Maryland
College Park, MD 20742
301.405.5433 TEL
301.314.9477 FAX
jwh2@umd.edu

1. INTRODUCTION

Equipment selection problems form a separate class of problems in the domain of manufacturing system design. Equipment selection determines the type and quantity of the machine tools (or other equipment) that will be installed at each workstation in a manufacturing system. Bretthauer [1] addresses capacity planning in manufacturing systems by modeling them as a network of queues. Assuming a single server at each node, a branch-and-bound algorithm is presented to find a minimum cost selection of capacity levels from a discrete set of choices, given a constraint on the WIP. Swaminathan [2] provides an analytical model for procurement of tools for a wafer fab incorporating uncertainties in the demand forecasts. The problem is modeled as a stochastic integer programming with recourse, and the objective is to minimize the expected stock-out costs due to lost sales across all demand scenarios. Considering only one tool type per workstation, the first stage variables - the number of tools procured, are decided before the demand occurs. The second stage variables determine the allocation of different wafer types to different tools in each demand scenario, after the demand is realized. Connors, Feigin and Yao [3] perform tool planning for a wafer fab using a queueing model, based on a marginal allocation procedure to determine the number of tools needed to achieve a target cycle time with the objective of minimizing overall equipment cost. Assuming identical tools at each tool group, their model incorporates detailed analysis of scrap and rework to capture the effects of variable job sizes on the workload and on the utilization of tool groups, and careful treatment of "incapacitation" events that disrupt the normal process at tools.

A flexible flow shop is a manufacturing system that consists of a sequence of workstations. Each workstation may have one or multiple resources (e.g., machine tools) working in parallel. This paper addresses equipment selection when, for each workstation in the flexible flow shop, there exist multiple (functionally identical) tool types available to purchase. The objective is to minimize the average manufacturing cycle time of jobs processed by the shop, subject to the constraints on the throughput and the budget available. (Note that the terms "flow time" and "throughput time" are sometimes used instead of "manufacturing cycle time.") Due to the complexity of this discrete optimization problem, an exact solution cannot be found in reasonable time. Thus, we present two versions of a search algorithm that uses the $M/M/m$ queueing system model to evaluate the average cycle time. This paper presents the algorithms and discusses the results of experiments conducted to evaluate the algorithms' performance across a wide range of problem instances.

The paper is organized as follows. Section 2 formulates the equipment selection problem. Section 3 describes the two analytical algorithms. Section 4 describes the experimental architecture used to evaluate the algorithms and discusses the results we obtained. Section 5 summarizes the paper.

2. PROBLEM FORMULATION

In many industries, especially semiconductor manufacturing, constructing new manufacturing facilities has grown increasingly expensive, as more sophisticated machine tools, cluster tools, and other equipment are required to produce high-quality goods. During the design of a manufacturing system, it is common to budget funds for purchasing equipment. In addition, the facility is built to achieve certain levels of production. Thus the manufacturing system must have sufficient capacity. The capacity of the workstations also strongly influences congestion and queueing during operation. Indeed, management would like to reduce manufacturing cycle time and work-in-process inventory as much as possible. All of these objectives are affected by the number and type of tools that are purchased for the manufacturing system.

This paper discusses the problem of selecting equipment (generically, tools) for a flexible flow shop. The shop will have n workstations. When the shop is operating, each job must visit all of the workstations in sequence, and the queue of jobs at each workstation will be first-in-first-out.

For workstation i, there are z_i types of tools available. The decision variables X_{ij} are the number of tools of type j purchased for each workstation i; $i = 1, ..., n$ and $j = 1, ..., z_i$. X_{ij} must be a non-negative integer. The cost of one tool of type j at workstation i is C_{ij} (dollars) and the capacity of one such tool is $_{ij}$ (jobs per unit time). The total number of discrete decision variables is p, where

$$p = \sum_{i=1}^{n} z_i$$

In practice, it is common to purchase identical tools for a workstation, which reduces training and maintenance costs. Thus, there are actually two decision variables for each workstation: which type of tool to purchase, and how many to purchase.

The objective is to minimize *CT*, the average manufacturing cycle time that jobs spend in the shop. (To evaluate the manufacturing cycle time, we use standard queueing system models, see, for example, [6].) The decision-maker has a fixed budget of *M* (dollars) for purchasing tools, which can be expressed in the following constraint:

$$\sum_{i=1}^{n}\sum_{j=1}^{z_i} X_{ij}C_{ij} \le M$$

Also, the manufacturing system must achieve a throughput of (jobs per unit time). Let ${}_i$ denote the capacity at workstation *i*.

$$\mu_i = \sum\nolimits_{j=1}^{z_i} X_{ij}\mu_{ij}$$

Since ${}_i$ must be greater than , then, for $i = 1, \ldots, n$,

$$\sum_{j=1}^{z_i} X_{ij}\mu_{ij} > \lambda$$

Note that for $n = 1$, this problem is equivalent to the integer knapsack problem, which has been shown to be NP-complete [5].

3. SOLUTION APPROACH

In general, the complexity of designing manufacturing systems arises due to the constraints on the system and the stochastic nature of the dynamics in the system. Simulation modeling can be an effective tool to model and evaluate such systems, especially when it is impossible (or difficult) to evaluate the objective function analytically. Simulation optimization techniques use simulation to solve stochastic optimization problems. However, simulation-based approaches require a large amount of computational effort. Thus, it is worth exploring analytical approaches in these situations.

The budget constraint and throughput constraints bound the set of feasible solutions. Purchasing too few tools will give insufficient capacity. Purchasing too many tools will violate the budget constraint.

Because the problem is NP-complete, we choose to pursue search algorithms that can explore the solution space and find near-optimal solutions. The first algorithm selects, for each workstation, the tools with the highest capacity. The second algorithm selects those tools that have the highest ratio of capacity to cost. Both algorithms proceed by creating an initial, feasible solution and then incrementally adding tools until either the budget constraint is tight or the improvement in solution quality is insignificant.

3.1 Notation

The notation used is as follows:

	desired throughput
M	budget available
n	number of workstations
z_i	total number of different tool types at workstation i; $i = 1, ..., n$
T_{ij}	tool of type j at workstation i; $j = 1, ..., z_i$
$_{ij}$	capacity of T_{ij} tool
C_{ij}	cost of T_{ij} tool
U_{ij}	capacity per unit cost of T_{ij} tool = $\frac{\mu_{ij}}{C_{ij}}$
k	iteration number
$\lfloor x \rfloor$	greatest integer less than or equal to x
$\lceil x \rceil$	smallest integer greater than or equal to x
T_i	selected tool type at workstation i
X_i	number of tools for workstation i

T_i and X_i are the decision variables. If $T_i = j$, $C_i = C_{ij}$ and $_i = {}_{ij}$.

the number of tools: $\{X_1, X_2, \ldots, X_n\}$

$f(\ _k)$ the manufacturing cycle time of the system given a solution $_k$ $\{X_{11}, X_{12}, \ldots, X_{1,z_1}; \ldots; X_{n1}, X_{n2}, \ldots, X_{n,z_n}\}$

3.2 Description

The two search algorithms are called Algorithm I (A-I) and Algorithm II (A-II). The only difference in the algorithms is the selection of T_i.

For Algorithm I (A-I), let $T_i = j$, such that $_{ij} > \ _{ik}$ for all $k \ \ z_i$ and $k \ \ j$. If $_{ij} = \ _{ik}$, then choose the tool type with lower cost. Set $C_i = C_{ij}$; $_i = \ _{ij}$.

For Algorithm II (A-II), let $T_i = j$, such that $U_{ij} > U_{ik}$ for all $k \ \ z_i$ and $k \ \ j$. If $U_{ij} = U_{ik}$, then choose the tool type with higher capacity. Set $C_i = C_{ij}$; $_i = \ _{ij}$.

After T_i are selected, each algorithm proceeds as follows:

Step 1: Check feasibility

For $i = 1, \ldots, n$:

$$X_i = \left\lceil \frac{\lambda}{\mu_i} \right\rceil$$

$$B = M - \sum_{i=1}^{n} X_i C_i$$

If $B < 0$, then return the solution as infeasible.

Else

Initialize $k = 0$

$_k = \{X_1, X_2, \ldots, X_n\}$

For i = 1, ..., n,

$$\rho_i = \frac{\lambda}{X_i \mu_i}$$

$$\pi_i = (1 + \sum_{l=1}^{X_i - 1} \frac{(X_i \rho_i)^l}{l!} + \frac{(X_i \rho_i)^{X_i}}{X_i!(1-\rho_i)})^{-1}$$

$$f(\theta_k) = \sum_{i=1}^{n} (\frac{1}{\mu_i} + \frac{1}{\mu_i} \frac{(X_i \rho_i)^{X_i} \pi_i}{X_i! X_i (1-\rho_i)^2})$$

Output $Cost_I = X_1 C_1 + \ldots + X_n C_n$ and $Cycle\ Time_I = f(\ _k)$

Step 2: Perform the search

Let $f(\ _{k-1})$ = .

Let   be a small positive number (in our experiments,   = 0.01 hours).

Define $P(B)$ = {i: C_i B} as the set of workstations with “affordable” tools (that is, the cost of a tool at any of these workstations is not greater than the unspent budget).

While $P(B)$ is not empty and $f(\ _k)$ $f(\ _{k-1})$ - , repeat the following loop:

Let i be the workstation in $P(B)$ that currently has the least capacity (the smallest value of $X_i\ _i$).

Update X_i, B, and k as follows: $X_i = X_i + 1$; $B = B - C_i$; $k = k + 1$.

$_k = \{X_1, X_2, \ldots, X_n\}$.

Calculate $f(\ _k)$.

Update $P(B)$.

If $f(\ _k) > f(\ _{k-1})$ - , then revise X_i, B, and k as follows: $X_i = X_i - 1$; $B = B + C_i$; $k = k - 1$.

Construct the solution   from $_k$ as follows:

for all i and j, $X_{ij} = X_i$ if $T_i = j$, and 0 otherwise.

Output $Cost_F = X_1C_1 + \ldots + X_nC_n$ and $Cycle\ Time_F = f(\ _k)$.

4. EXPERIMENTS

The purpose of the experiments is to compare the two algorithms over a range of problem instances and determine how the characteristics of the problem instances affect the algorithms' performance. Thus, these instances are not based on any specific problems from industrial applications. The data sets can be found on the World-Wide Web at the following website: http://www.isr.umd.edu/Labs/CIM/projects/mfgsys/index.html

4.1 Experimental Design

The experiments were run on 16 data sets with 10 instances per data set. Hence there are 160 problem instances. In all of the instances, the flexible flow shop has five workstations. The desired throughput equals four jobs per hour. That is, the mean job interarrival time equals 0.25 hours. The job interarrival times and the job processing times are exponentially distributed. The mean processing time on a tool of type j at workstation i is $1/\ _{ij}$. The tool capacity includes any detractors due to setups, rework, or failures. Travel times are ignored (in practice, these will be determined by the layout of the shop, which is not considered here).

To generate the data sets, we used the following parameter values:

P = cost factor for tool types = \$1000
= desired throughput = 4 jobs per hour
n = number of workstations = 5
r = expected number of tools per workstation = 2 or 10
z_i = number of tool types per workstation = 2 or 5
e = shape of correlation = 0.5 or 1
= lower bound of cost range = 0.5
= multiplier for budget = 1 or 3

The parameters r, z_i, e and   can take two values. Each combination of these values forms one of the sixteen data sets. Each instance in a data set was generated as follows:

$M = \ nrP$

For $i = 1, \ldots, n$,

For $j = 1, \ldots, z_i$,

Choose b_{ij} randomly from a uniform distribution over the range [, 1].

$a_{ij} = 2(b_{ij})^{e}$

$_{ij} = a_{ij}(\ /r)$

$C_{ij} = b_{ij}P$

4.2 Experimental Results

Each search algorithm (A-I and A-II) was run on each instance. The output of each run included five performance measures. The performance measures of the initial solution are $Cost_I$ and $Cycle\ Time_I$. The performance measures of the final solution are $Cost_F$ and $Cycle\ Time_F$. Since each data set is different, we normalized these statistics by comparing the cost performance to the total budget for that data set and comparing the cycle time performance to the expected total processing time of that data set. If b has a uniform distribution over $[l,u]$, then the expected value of $b^{0.5}$ can be calculated as follows:

$$E[b^{0.5}] = \frac{2}{3}\left(\frac{u^{1.5} - l^{1.5}}{u - l}\right)$$

From these statistics, the following performance metrics are calculated to estimate the performance of each algorithm on each instance:

$Cost\ Metric_I = Cost_I/M$. $Cost\ Metric_F = Cost_F/M$.

$Cycle\ Time\ Metric_I = Cycle\ Time_I/1.450$ and $Cycle\ Time\ Metric_F = Cycle\ Time_F/1.450$ if $e = 0.5$ and $r = 2$ (Data sets 1, 2, 9, and 10).

$Cycle\ Time\ Metric_I = Cycle\ Time_I/7.246$ and $Cycle\ Time\ Metric_F = Cycle\ Time_F/7.246$ if $e = 0.5$ and $r = 10$ (Data sets 5, 6, 13, and 14).

$Cycle\ Time\ Metric_I = Cycle\ Time_I/1.667$ and $Cycle\ Time\ Metric_F = Cycle\ Time_F/1.667$ if $e = 1.0$ and $r = 2$ (Data sets 3, 4, 11, and 12).

$Cycle\ Time\ Metric_I = Cycle\ Time_I/8.333$ and $Cycle\ Time\ Metric_F = Cycle\ Time_F/8.333$ if $e = 1.0$ and $r = 10$ (Data sets 7, 8, 15, and 16).

The fifth performance measure was the number of iterations that the algorithm performed before stopping. All of the metrics were averaged over all ten problem instances. Table 1 shows the results for each algorithm on each data set. Figures 1 and 2 also display the cost and cycle time metrics. A larger cost metric implies that more of the budget was spent purchasing tools. A larger cycle time metric implies that jobs spent more time in the system.

4.3 Summary of Results

The last two columns in Table 1 show that the number of iterations for both algorithms is approximately the same in most data sets. A-II does require more iterations in some data sets. The most significant increases occur in data sets 9 and 11 because A-I selects, in general, more expensive tools and spends the budget more quickly than A-II.

As shown in Table 1 and Figures 1 and 2, A-I constructs initial solutions that have, in general, a larger cost metric and a smaller cycle time metric than the initial solutions that A-II constructs. This results from A-I's selection of large capacity tools, which are expensive. But the initial solution is likely to have more than enough capacity, which reduces congestion and cycle time. A-II selects, in general, smaller tools, so the capacity of the initial solution will exceed the throughput constraint by a smaller margin. Higher utilization will lead to larger cycle times.

At the end of the search, A-I finds solutions that have a larger cost metric than the final solutions that A-II finds, but the performance on the cycle time metric is very close. Compared to the initial solutions, the final solutions found have much larger cost metrics and much smaller cycle time metrics. Thus, it is clear that the search algorithms are useful for finding feasible, high-quality solutions.

The impact of the budget constraint is significant. Figure 1 shows that, in odd-numbered data sets (where = 1), both search algorithms spend most of the budget by the time the search ends. However, in even-numbered data sets (where = 3), the search algorithms leave much of the budget unspent.

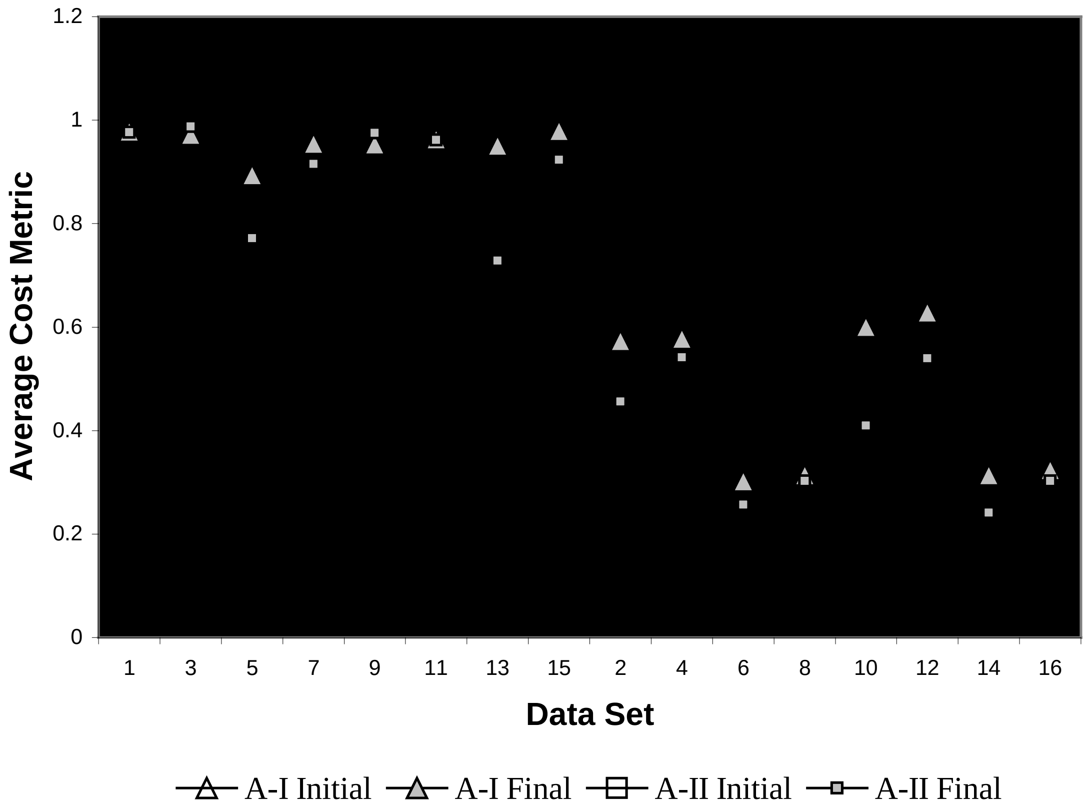

Figure 1: Average cost metric

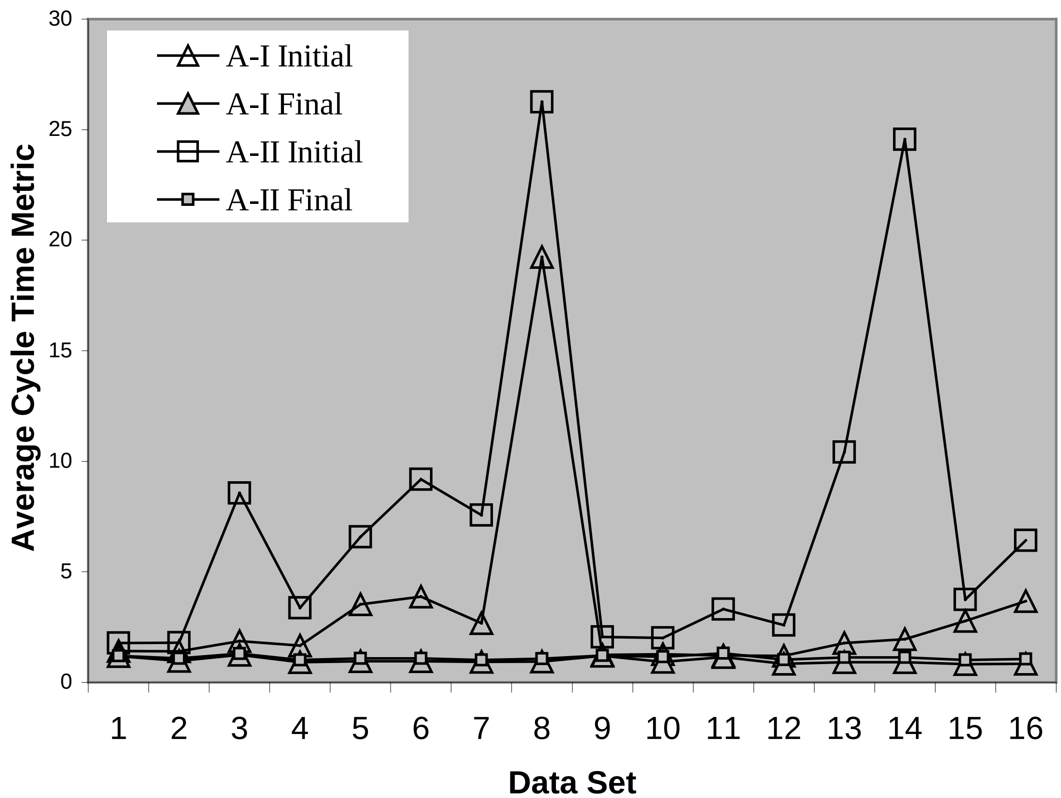

Figure 2: Average cycle time metric

Data set	n	z	r	e	β	*Cost Metric*$_I$		*Cost Metric*$_F$		*Cycle Time Metric*$_I$		*Cycle Time Metric*$_F$		Number of Iterations	
						A-I	A-II	A-I	A-II	A-I	A-II	A-I	A-II	A-I	A-II
1	5	2	2	0.5	1	0.823	0.666	0.979	0.976	1.420	1.783	1.190	1.196	2.3	5.0
2	5	2	2	0.5	3	0.277	0.219	0.574	0.456	1.403	1.792	0.966	1.091	10.9	11.0
3	5	2	2	1.0	1	0.809	0.759	0.972	0.988	1.861	8.565	1.247	1.298	2.4	3.6
4	5	2	2	1.0	3	0.285	0.262	0.578	0.542	1.668	3.368	0.915	1.007	10.6	11.2
5	5	2	10	0.5	1	0.504	0.433	0.895	0.772	3.532	6.581	0.961	1.084	23.9	26.3
6	5	2	10	0.5	3	0.173	0.146	0.303	0.257	3.877	9.190	0.948	1.079	23.2	25.5
7	5	2	10	1.0	1	0.553	0.543	0.955	0.916	2.679	7.571	0.923	1.018	24.6	25.2
8	5	2	10	1.0	3	0.181	0.178	0.314	0.303	19.244	26.257	0.944	1.067	25.0	26.4
9	5	5	2	0.5	1	0.916	0.577	0.954	0.975	1.251	2.051	1.207	1.214	0.5	7.1
10	5	5	2	0.5	3	0.302	0.196	0.601	0.410	1.267	2.015	0.924	1.152	10.0	11.0
11	5	5	2	1.0	1	0.908	0.769	0.964	0.961	1.215	3.306	1.149	1.309	0.7	3.0
12	5	5	2	1.0	3	0.305	0.258	0.629	0.540	1.205	2.584	0.836	1.025	10.7	11.6
13	5	5	10	0.5	1	0.552	0.412	0.951	0.728	1.779	10.419	0.908	1.134	21.8	26.9
14	5	5	10	0.5	3	0.181	0.136	0.314	0.241	1.947	24.565	0.916	1.134	22.2	26.9
15	5	5	10	1.0	1	0.559	0.546	0.980	0.923	2.775	3.753	0.824	1.013	23.0	25.2
16	5	5	10	1.0	3	0.186	0.179	0.324	0.302	3.667	6.429	0.842	1.049	23.1	25.7

Table 1: Results for the problem set

5. SUMMARY AND CONCLUSIONS

This paper presented an equipment selection problem that seeks to minimize the average manufacturing cycle time subject to two constraints: the amount of funds available to purchase equipment is limited, and the capacity of the manufacturing system must be larger than the desired throughput. Similar to Cochran *et al*. [7], this paper emphasizes that equipment selection should be done by examining how various solutions affect manufacturing system performance, not just per unit cost of the operation. This paper presented two search algorithms for the problem and presented results of experiments designed to show how they perform across a wide range of problem instances.

It is worth noting that the equipment selection problem we considered has a special structure to it. It seems intuitive that given a choice between a variety of tools, the addition of a higher capacity tool will serve to reduce the cycle time more. Moreover, an even distribution of the capacity of workstations tends to avoid serious bottlenecks that occur when the capacity distribution is skewed. The search algorithms presented here exploit this special structure. However, it may be inappropriate for more complex manufacturing systems such as job shops where different workstations have different throughput requirements. If the interarrival and processing times have other probability distributions, a more general $GI/G/m$ approximation would be required to estimate manufacturing cycle times. See Herrmann and Chincholkar [8] for instance.

It will require more effort to extend this approach to equipment selection in more complex job shops and to systems that will use kanban or CONWIP production control mechanisms. In such systems, open queueing networks (which presume a push production control philosophy) are not appropriate. Existing analytical models become less accurate and the need for discrete-event simulation grows. For example, Kumar [4] developed simulation-based optimization algorithms that could be extended to other settings.

Acknowledgement

The Semiconductor Research Corporation and the National Science Foundation under grant DMI 9713720 supported this research conducted in the facilities of the Computer Integrated Manufacturing Lab at University of Maryland, College Park. Jeffrey W. Herrmann has a joint appointment with the Department of Mechanical Engineering. The authors appreciate the help and the useful comments provided by Dr. Michael Fu, Dr. Steven Marcus and Dr. Gary Rubloff.

References

[1] Bretthauer, K.M., "Capacity planning in manufacturing and computer networks," *European Journal of Operational Research*, Vol.91, pp.386-394, 1996.

[2] Swaminathan, J.M., "Tool capacity planning for semiconductor fabrication facilities under demand uncertainty," *European Journal of Operational Research*, Vol.120, pp.545-558, 2000.

[3] Connors, D.P., G.E.Feigin and D.D.Yao, "A queueing network model for semiconductor manufacturing," *IEEE Transasctions on Semiconductor Manufacturing*, Vol.9, No.3, pp.412-427, 1996.

[4] Kumar, R., "Simulation optimization for manufacturing system design," M.S. Thesis, Institute for Systems Research, University of Maryland, College Park, 2002.

[5] Garey, M.R. and D.S.Johnson, *Computers and Intractability, a guide to the theory of NP-Completeness*, W.H.Freeman and Company, San Francisco, CA, pp.247, 1979.

[6] Hall, R.W., *Queueing methods for services and manufacturing*, Prentice Hall, Englewood Cliffs, NJ, pp.143, 1991.

[7] Cochran, D.S., J.F.Arinez, F.W.Duda and J.Linck, "A decomposition approach for manufacturing system design," *Journal of Manufacturing Systems*, Vol.20, No.6, pp.371-389, 2001/2002.

[8] Herrmann, J.W. and M.M.Chincholkar, "Reducing throughput time during product design," *Journal of Manufacturing Systems*, Vol.20, No.6, pp.416-428, 2001/2002.f