European Journal of Operational Research 45 (1990) 47–54
North-Holland

Machine allocation problems in manufacturing networks

O.J. BOXMA
Centre for Mathematics and Computer Science, Amsterdam, The Netherlands, and Faculty of Economics, Tilburg University, The Netherlands

A.H.G. RINNOOY KAN
Econometric Institute, Erasmus University Rotterdam, The Netherlands

M. van VLIET
Tinbergen Institute, Erasmus University Rotterdam, The Netherlands, and AKB consultants, Rotterdam, The Netherlands

Abstract: In this paper we discuss two server (machine) allocation problems that occur in manufacturing networks. The manufacturing network is modelled as an open network of queues. The server allocation problems are solved by means of a marginal analysis scheme. We show that for the first problem our algorithm generates undominated allocations. Furthermore, the algorithm provides us with bounds to check how close the allocation generated is to the optimal allocation. In the second problem the algorithm presented generates optimal allocations within time bounded by a polynomial function in the size of the network.

Keywords: Combinatorial optimization, flexible manufacturing systems, manufacturing networks, marginal analysis, server allocation, queueing network

1. Introduction

Manufacturing operations tend to become more complex over time. This increasing complexity makes it more and more difficult for management to understand how their manufacturing plant operates. In particular, the relationship between performance criteria such as work-in-process (WIP), lead times, costs, investment in capital etc. becomes very complicated. This calls for a more thorough understanding of such relationships. This paper tries to give an answer to one such problem that occurs in manufacturing today.

We shall discuss problems dealing with the *design* of manufacturing networks. Design issues naturally emerge on a midterm to long-term time horizon, involving decisions on a strategic management level. The problems we consider represent two important issues. The first issue concerns the design of a manufacturing network such that the network satisfies a certain performance level (e.g. WIP, lead times). The costs, however, related with the design have to be minimal. One can think, for example, of a plant manager who wants the average lead time of the products not to exceed four weeks. The problem is then to design the network such that this target lead time is met, while keeping costs as low as possible.

The second issue concerns problems where a fixed amount of machinery has to be distributed across the network. Here the aim is to make optimal (in terms of network performance) use of the machinery. Problems like this often emerge in designing flexible manufacturing systems. Imagine, e.g., the situation in which a fixed number of homogeneous machines is available. Each machine can be made suitable for any kind of operation by

Revised June 1989

assigning different tools to the machine. The problem now is to distribute the available machines over the network so as to optimize the performance of the network.

In the sequel we assume that product routings, throughput of products, location of machines, and technology are already specified. The design issues we address are capacity issues; how many machines do we assign to each location?

Several authors (e.g. Bitran and Tirupati, 1987, 1988; Buzacott and Yao, 1986; Whitt, 1983) stress the fact that queueing network theory provides an excellent means for evaluating designs of manufacturing networks. We show that by combining queueing network theory with combinatorial optimization techniques we not only are able to evaluate designs but also to give quantitative answers to questions of optimal design.

We consider a production process that can be modelled as an open network of queues with different product classes. In our context this means that the production process consists of several workstations through which each product follows its own individual deterministic route. A workstation consists of several parallel identical machines (servers). An example of such a network is a production process where printed circuit boards are made. A certain type of circuit board will only visit a workstation if that particular workstation prints a component that belongs to that type of circuit board. Furthermore, if different types of circuit boards use the same type of component, they will all visit the workstation that prints that component.

In this paper we will study problems concerning the optimal allocation of servers to workstations. In particular, given product mix, throughput, and technology, we aim to optimally allocate servers such that either

(a) the WIP level of the network satisfies a certain target WIP level and the costs of the configuration are minimal (this problem will be called the "server allocation problem" (SA)), or

(b) the WIP level is minimized while keeping the total number of servers fixed (in the sequel we will call this problem the "server reallocation problem" (SR)).

Since WIP and lead times are linearly related through Little's law (Little, 1960) the problems we discuss and the algorithms to solve them also relate to the latter performance measure.

The algoritms we present are based on the marginal analysis approach as developed by Fox (1966) and later used by Rolfe (1971), Weinstein and Yu (1973), and Weber (1980). The marginal analysis method is in fact a greedy method. It starts with a non-feasible allocation and it adds servers to workstations where the best local improvement is achieved. The algorithm terminates when an allocation becomes feasible.

In the present study it is assumed that all interarrival times and service times of the manufacturing network are exponentially distributed. This assumption allows a detailed analysis of the queueing model describing the manufacturing network. In actual manufacturing networks the exponentiality assumptions are often not realistic. A common way to reduce work-in-process is by splitting up processing tasks such that more or less equal portions are obtained for different product types. This leads to service times which have much smaller coefficients of variation than exponential service times. In a future study (van Vliet and Rinnooy Kan, 1989) we extend the present approach to more general arrival and service processes building upon concepts developed in this paper.

The organization of this paper is as follows. In Section 2 we give a brief review on what has been reported on related problems in the literature. In Section 3 the open queueing network under consideration is described and an exact expression for the mean queue length at each queue is presented. The server allocation problem and reallocation problem are discussed in respectively Sections 4 and 5. In these sections we present the appropriate mathematical models as well as algorithms to solve these problems. Moreover, we will elaborate on the quality of the solutions generated by the algorithm. Conclusions and suggestions for further research are stated in Section 6.

2. Review of the literature

An extensive literature exists on the optimal configuration of queueing networks. Early work concentrated on design issues concerning computer and telecommunication networks. Recently, much research has been conducted on planning issues in manufacturing networks (flexible manufacturing systems, assembly systems etc.). We con-

centrate on the problem of assigning capacity, given the network topology and given arrival and service processes. We distinguish between single server networks and multi server networks; in single server networks assignment of capacity is translated into allocation of service rates, while in multi server networks assignment of capacity is translated into allocation of numbers of servers.

Gerla and Kleinrock (1977), and Kleinrock (1976) (Chapters 5 and 6) considered several design issues of computer communication networks in which each workstation is modelled as an M/M/1 queue. One of the problems they address is the capacity assignment problem, i.e. the problem of how to assign capacity so as to minimize a certain cost function. Here capacity represents the numbers of bits per second that can be transmitted over a channel and is expressed in service rates. This capacity assignment problem is clearly related to our (SA) problem, but in the latter one the optimal solution must be selected among a discrete set of possibilities.

Motivated by design problems in manufacturing networks, Bitran and Tirupati (1987) have recently studied a network in which each workstation can be modelled as a GI/G/1 queue. They considered two capacity assignment problems (again, capacity is expressed in terms of service rates). The first problem, the 'targeting problem', addresses the issue of capacity assignment to workstations in order to meet a target-WIP level while attaining minimal costs. The other problem, the 'balancing problem', concerns the division of available capacity among the workstations so as to minimize the WIP of the network.

In manufacturing networks the assignment of capacity often amounts to the assignment of machines to workstations. In queueing terminology, this is the assignment of numbers of servers to the service stations of the queueing network. Dallery and Frein (1986), and Shantikumar and Yao (1987) considered server allocation issues in closed queueing networks. Dallerey and Frein are concerned with minimal cost server allocation in order to achieve a given production rate. They present marginal analysis algorithms to heuristically solve the problems they discuss. Since no specific properties are assumed for the objective functions, they cannot give any theoretical results on the quality of their heuristics. They do perform several numerical experiments and show that their heuristic solution are very close (or equal) to the optimal solutions. Shantikumar and Yao (1987) consider a closed queueing network with fixed buffer capacities (maximum total number of products at a workstation). For this network they discuss the problem of dividing a fixed total number of servers among the workstations so as to minimize a certain profit function. Although the queueing network they discuss is different from the queueing network we consider, their problem is clearly related to our (SR) problem. They show that a marginal analysis scheme generates an optimal solution.

3. Analysis of the network

The manufacturing network we consider consists of J workstations. Each workstation j has m_j identical parallel servers with independent exponentially distributed service times with mean $1/\mu_j$. N product types are produced by the network. Products of type i arrive at the first workstation they visit according to a Poisson process with parameter λ^i, and then follow a deterministic route through a subset of the set of workstations. A product may visit a workstation more than once, but for simplicity we shall not allow two successive stages of a product route to be identical. Furthermore, it is assumed that the arrival processes and service processes are independent.

For further analysis we can treat the different product types as one aggregate product with an aggregate arrival rate λ_j at each workstation j. The joint equilibrium queue length distribution in the network has a product form (cf. Kelly, 1979, Corollary 3.4). In the steady state each workstation j behaves as an $M/M/m_j$ queue. This leads to the following well-known formula for the average number of products present (in queue and in process) at workstation j (cf. Tijms, 1986, p. 332):

$$L_j(m_j, \mu_j, \lambda_j) = \frac{(\lambda_j/\mu_j)^{m_j}(\lambda_j/(\mu_j m_j))}{m_j!(1-\lambda_j/(\mu_j m_j))^2} \times \left\{ \sum_{k=0}^{m_j-1} \frac{(\lambda_j/\mu_j)^k}{k!} + \frac{(\lambda_j/\mu_j)^{m_j}}{m_j!(1-\lambda_j/(\mu_j m_j))} \right\}^{-1} + \frac{\lambda_j}{\mu_j}.$$

In the sequel we assume that the arrival and service rates are given, while the numbers of servers at workstations are decision variables. This means that $L_j(m_j, \mu_j, \lambda_j)$ can be regarded as a function of m_j only: $L_j(m_j)$. Dyer and Proll (1977) have proved that $L_j(m_j)$ is a convex, decreasing, function in m_j.

We will measure the steady state performance of the network by the WIP of the network. The WIP (inventory) is the total value of all the products that are in the network. Without loss of generality, we make the assumption that the value of a product at workstation j, either in queue or in process, is independent of the type of product and equal to v_j. The formulation for WIP then becomes

$$\mathrm{WIP}(m_1, \ldots, m_J) = \sum_{j=1}^{J} v_j L_j(m_j).$$

Furthermore, we assume that the allocation of m_j servers at workstation j generates investment costs of $F_j(m_j)$ with $F_j(m_j)$ a convex and non-decreasing function in m_j. Such functions are of interest since they can model situations where capacity increments are achieved by using cheaper options initially. Bitran and Tirupati (1987) also mention that with regard to over-time wage structure, convex investment functions are a proper representation of these decision problems.

In order to prevent the system from becoming instable, we have to require that the traffic intensity at a workstation j $(=\lambda_j/(m_j\mu_j))$ is less than one. It is easy to verify that this results in requiring that $m_j \geqslant m_j^{\mathrm{L}} = \lfloor \lambda_j/\mu_j \rfloor + 1$, where $\lfloor\ \rfloor$ represents the integer rounddown operation. For convenience we will use the following notation:

$$m = (m_1, \ldots, m_J), \qquad S = \{ m \mid m_j \geqslant m_j^{\mathrm{L}} \},$$

$$F(m) = \sum_{j=1}^{J} F_j(m_j), \qquad L(m) = \sum_{j=1}^{J} v_j L_j(m_j),$$

$$\Delta F_j(m_j) = F_j(m_j) - F_j(m_j - 1),$$

$$\Delta L_j(m_j) = L_j(m_j) - L_j(m_j - 1).$$

From the convexity of F_j and L_j $(j \in \{1, \ldots, J\})$ it follows that

$$\frac{\Delta F_j(m_j + 1)}{-v_j \Delta L_j(m_j + 1)} \geqslant \frac{\Delta F_j(m_j)}{-v_j \Delta L_j(m_j)}, \tag{3.1}$$

and

$$v_j \Delta L_j(m_j + 1) \geqslant v_j \Delta L_j(m_j). \tag{3.2}$$

4. The server allocation problem

In this problem we want to allocate servers in such a way that the WIP is below a target WIP level W_{T}. The configuration we are looking for is a minimal cost configuration: the investment costs associated with allocating the servers have to be minimal. The mathematical formulation is as follows:

$$\begin{array}{lll} \text{(SA)} & \text{Minimize} & F(m) \\ & \text{subject to} & L(m) \leqslant W_{\mathrm{T}}, \\ & & m_j \geqslant m_j^{\mathrm{L}}, \quad m_j \text{ integer} \\ & & (j \in \{1, \ldots, J\}). \end{array}$$

The algorithm to solve (SA) is a very natural one. It starts with the smallest possible allocation, that is m_j^{L} for each workstation j. At every iteration it then adds a server at that workstation where the quotient of the increase of the objective function and the decrease of the work-in-process is the smallest. The algorithm terminates as soon as adding a server makes the allocation feasible.

Algorithm 1

1. Start with c^0 where $c_j^0 = m_j^{\mathrm{L}}$.
2. $k := 1$.
3. Set $c^k := = c^{k-1} + e_i$, where e_i is the i-th unit vector and i is an index for which

$$\frac{\Delta F_j(c_j^{k-1} + 1)}{-v_j \Delta L_j(c_j^{k-1} + 1)} \quad (j \in \{1, \ldots, J\})$$

is minimal.

4. If $L(c^k) \leqslant W_{\mathrm{T}}$, stop; else $k := k + 1$, go to Step 3.

In the following we will analyse how close the allocation generated by Algorithm 1 is to the optimal allocation.

An allocation x is called *undominated* (efficient) (cf. Fox, 1966) if for all $y \in S$,

$$F(y) < F(x) \Rightarrow L(y) > L(x),$$

$$F(y) = F(x) \Rightarrow L(y) \geqslant L(x).$$

We will show that at each iteration of the algorithm an undominated allocation is generated.

Lemma 1. *If $\tau \geqslant 0$ and $x^* \in S$ minimizes $F(x) + \tau L(x)$ for all $x \in S$, then x^* minimizes $F(x)$ over all $x \in S$ for which $L(x) \leqslant L(x^*)$.*

Proof. Suppose $x \in S$ for which $L(x) \leqslant L(x^*)$, then $\tau L(x) - \tau L(x^*) \leqslant 0$. However, from the fact that x^* minimizes $F(x) + \tau L(x)$ it follows that $F(x^*) - F(x) \leqslant \tau L(x) - \tau L(x^*)$. Hence, $F(x^*) \leqslant F(x)$ for all $x \in S$ for which $L(x) \leqslant L(x^*)$. □

Let $m_j^*(\tau)$ be the smallest integer $m \geqslant m_j^L$ such that

$$\Delta F_j(m+1) > \tau\big(-v_j \Delta L_j(m+1)\big),$$

and let $M_j^*(\tau)$ be the (possibly empty) set of integers $m \geqslant m_j^L$ such that

$$\Delta F_j(m+1) = \tau\big(-v_j \Delta L_j(m+1)\big),$$

and define $M_j(\tau) = m_j^*(\tau) \cup \{M_j^*(\tau)\}$, and $M(\tau) = \otimes_{j=1}^{J} M_j(\tau)$.

Since $F(x)$ and $L(x)$ are respectively convex increasing and convex decreasing functions in x, $M(\tau)$ is the set of global minima of $F(x) + \tau L(x)$ for all $x \in S$. Furthermore, from Lemma 1 we get the following corollary.

Corollary 1. *$x(\tau) \in M(\tau) \Rightarrow x(\tau)$ is undominated.*

We can now prove the following theorem.

Theorem 1. *Allocations generated by Algorithm 1 are undominated.*

Proof. c^0 is clearly undominated. We now set τ_k equal to the minimum of Step 3 in the $(k+1)$st iteration of the algorithm. By (3.1) we know that $\tau_k \geqslant \tau_{k-1}$. We now use induction with k as index. By definition of τ_1 we know that $c^1 \in M(\tau_1)$. Suppose that $c_j^{k-1} \in M_j(\tau_{k-1})$ $(j \in \{1, \ldots, J\})$. By definition of τ_k we know that

$$\Delta F_j\big(c_j^k + 1\big) \geqslant \tau_k\big(-v_j \Delta L_j\big(c_j^k + 1\big)\big). \tag{4.1}$$

From the induction hypothesis the following two possibilities can occur:

(i)

$$\Delta F_j\big(c_j^{k-1} + 1\big) = \tau_{k-1}\big(-v_j \Delta L_j\big(c_j^{k-1} + 1\big)\big).$$

For every integer $m \geqslant m_j^L$ smaller than c_j^k we then know, from (3.1), that

$$\Delta F_j(m+1) \leqslant \tau_{k-1}\big(-v_j \Delta L_j(m+1)\big).$$

From (4.1) it then follows that c_j^k is either the smallest integer $m \geqslant m_j^L$ satisfying

$$\Delta F_j(m+1) > \tau_k\big(-v_j \Delta L_j(m+1)\big),$$

or c_j^k satisfies

$$\Delta F_j\big(c_j^k + 1\big) = \tau_k\big(-v_j \Delta L_j\big(c_j^k + 1\big)\big).$$

Hence, $c_j^k \in M_j(\tau_k)$.

(ii) c_j^{k-1} is the smallest integer $m \geqslant m_j^L$ satisfying

$$\Delta F_j(m+1) > \tau_{k-1}\big(-v_j \Delta L_j(m+1)\big).$$

We now have to distinguish between two cases.

(a) $c_j^k = c_j^{k-1}$, and

(b) $c_j^k = c_j^{k-1} + 1$.

Ad (a). If $c_j^k = c_j^{k-1}$ then by (4.1) and $\tau_k \geqslant \tau_{k-1}$ it follows that c_j^k is either the smallest integer $m \geqslant m_j^L$ satisfying

$$\Delta F_j(m+1) > \tau_k\big(-v_j \Delta L_j(m+1)\big),$$

or c_j^k satisfies

$$\Delta F_j\big(c_j^k + 1\big) = \tau_k\big(-v_j \Delta L_j\big(c_j^k + 1\big)\big).$$

Hence, $c_j^k \in M_j(\tau_k)$.

Ad (b). If $c_j^k = c_j^{k-1} + 1$ then by Algorithm 1,

$$\Delta F_j\big(c_j^{k-1} + 1\big) = \tau_k\big(-v_j \Delta L_j\big(c_j^{k-1} + 1\big)\big).$$

From (3.1) we know that for every integer $m \geqslant m_j^L$ smaller than c_j^k,

$$\Delta F_j(m+1) \leqslant \tau_k\big(-v_j \Delta L_j(m+1)\big).$$

The same argument as in (i) now leads to $c_j^k \in M_j(\tau_k)$.

So we see that $c_j^{k-1} \in M_j(\tau_{k-1})$ implies that $c_j^k \in M_j(\tau_k)$. Now apply Corollary 1. □

Theorem 1 shows that the allocations generated by Algorithm 1 are undominated. This does not necessarily imply that the allocation with which the algorithm terminates is also an optimal allocation. However, we can prove the following theorem.

Theorem 2. *If $c^0, \ldots, c^p$ are the allocations generated by Algorithm* 1 *and c^* is an optimal allocation for* (SA) *then it holds that*

$$F(c^{p-1}) < F(c^*) \leqslant F(c^p).$$

Proof. c^* is an optimal allocation, hence $F(c^*) \leqslant F(c^p)$. Since by definition of Algorithm 1, allocation c^{p-1} is infeasible it follows that $L(c^{p-1}) > L(c^*)$. Since c^{p-1} is also undominated, $L(c^{p-1}) > L(c^*)$ implies that $F(c^{p-1}) < F(c^*)$. □

Hence, the solution generated by Algorithm 1 provides us with bounds to check whether the allocation found by Algorithm 1 is sufficiently close to the optimal allocation. In a follow-up study (van Vliet and Rinnooy Kan, 1989) we report on numerical results of the (SA) algorithm for two real-life manufacturing networks. These numerical results show that the upperbound on the relative error of the allocation resulting from the (SA) algorithm for different target-WIP values is around 5%. This suggests that the allocation generated by the (SA) algorithm is, for most real-life applications, sufficiently close to the optimal allocation.

5. The server reallocation problem

The server reallocation problem is the problem of allocating servers to workstations such that the WIP is minimized. The number of servers that can be allocated is fixed and equal to M. The allocation of servers can be regarded as the distribution of these M servers over the queueing network. The mathematical formulation is as follows.

Server reallocation

$$\begin{aligned} \text{(SR)} \quad \text{Minimize} \quad & L(m) \\ \text{subject to} \quad & \sum_{j=1}^{J} m_j = M, \\ & m_j \geqslant m_j^{\mathrm{L}}, \quad m_j \text{ integer} \\ & (j \in \{1, \ldots, J\}). \end{aligned}$$

The algorithm starts with the smallest possible allocation. At each iteration a server is added to that workstation where the greatest decrease in WIP is achieved. This is being repeated until all the servers from the pool of M servers have been allocated.

Algorithm 2

1. Start with c^0 where $c_j^0 = m_j^{\mathrm{L}}$.
2. $k := 1$.
3. Set $c^k = c^{k-1} + e_i$, where e_i is the i-th unit vector and i is an index for which

$$v_j \Delta L_j\left(c_j^{k-1} + 1\right) \quad (j \in \{1, \ldots, J\})$$

is minimal.

4. If $k = M - \sum_{j=1}^{J} m_j^{\mathrm{L}}$ stop; else $k := k + 1$, go to Step 3.

Theorem 3. *The allocation c^* at which Algorithm* 2 *terminates is an optimal allocation for* (SR).

Proof. Suppose there is an allocation c with $\sum_{j=1}^{J} c_j = M$ and $L(c) < L(c^*)$. We can then transform c^* into c by performing a (minimal) number of permutations of one server from one workstation to another workstation. Suppose that at a certain permutation the number of servers z_{b} at workstation b is decreased by one while the number of servers z_{d} at workstation d is increased by one.

We want to show that by performing the permutation the objective function does not decrease. This means that we have to prove that

$$v_{\mathrm{d}} \Delta L_{\mathrm{d}}(z_{\mathrm{d}} + 1) \geqslant v_{\mathrm{b}} \Delta L_{\mathrm{b}}(z_{\mathrm{b}}). \tag{5.1}$$

Since we do a minimal number of permutations we know that $z_{\mathrm{b}} \leqslant c_{\mathrm{b}}^*$ and $z_{\mathrm{d}} \geqslant c_{\mathrm{d}}^*$. From (3.2) we then get

$$v_{\mathrm{d}} \Delta L_{\mathrm{d}}(z_{\mathrm{d}} + 1) \geqslant v_{\mathrm{d}} \Delta L_{\mathrm{d}}(c_{\mathrm{d}}^* + 1), \tag{5.2}$$

and

$$v_{\mathrm{b}} \Delta L_{\mathrm{b}}(c_{\mathrm{b}}^*) \geqslant v_{\mathrm{b}} \Delta L_{\mathrm{b}}(z_{\mathrm{b}}). \tag{5.3}$$

From the algorithm it follows that

$$v_{\mathrm{d}} \Delta L_{\mathrm{d}}(c_{\mathrm{d}}^* + 1) \geqslant v_{\mathrm{b}} \Delta L_{\mathrm{b}}(c_{\mathrm{b}}^*). \tag{5.4}$$

Inequality (5.4) can beverified by considering the iteration k of the algorithm at which the number of servers at workstation b is increased from $c_{\mathrm{b}}^* - 1$ to c_{b}^*. c_{d}^k is the number of servers of workstation d at iteration k. It then holds that $c_{\mathrm{d}}^k \leqslant c_{\mathrm{d}}^*$. From the algorithm we also know, since $c_{\mathrm{b}}^* - 1$ is in-

creased by one, that $v_b \Delta L_b(c_b^*)$ is the minimum attained at iteration k. Hence

$$v_d \Delta L_d(c_d^k + 1) \geq v_b \Delta L_b(c_b^*). \quad (5.5)$$

From (3.2) and $c_d^k \leq c_d^*$ it follows that

$$v_d \Delta L_d(c_d^* + 1) \geq v_d \Delta L_d(c_d^k + 1). \quad (5.6)$$

Inequalities (5.5) and (5.6) now give us (5.4). By combining (5.2), (5.3), and (5.4) we get

$$v_d \Delta L_d(z_d + 1) \geq v_b \Delta L_b(z_b),$$

which proves (5.1).

Since the numbers of servers of workstations $j \notin \{b, d\}$ at that particular permutation do not change, we see from (5.1) that at each permutation the objective function of (SR) does not decrease. Hence, $L(c) \geq L(c^*)$. So the minimal objective function is indeed achieved at c^*. □

It is easy to verify that algorithm 2 solves (SR) in $O(M * J)$ steps of the algorithm.

6. Conclusions and suggestions for further research

We have shown that a marginal analysis scheme provides a good environment for solving the two presented server allocation problems. For the (SR) problem the marginal analysis algorithm optimally solves the problem in $O(M * J)$ time. The algorithm presented for the (SA) problem generates an undominated allocation. Furthermore, the final allocation enables us to check how close the allocation is to the optimal allocation.

In the machine allocation problems considered, we did not restrict the interchangeability of the machines. In practice this interchangeability is often restricted to certain subclasses of operations. This means that machines can only be (re)allocated within a group of workstations that perform an operation belonging to a specific subclass. These kind of restrictions pose a different class of allocation problems creating interesting optimization problems for future research.

In this study all interarrival times and service times are exponentially distributed. In order to study more realistic manufacturing networks, we have recently studied the extension of the marginal analysis scheme to GI/G/m networks (see van Vliet and Rinnooy Kan, 1989). For GI/G/m networks it is not possible to give an exact analysis of the steady state behavior of the network. Hence, we hae to rely on techniques to approximate the steady state behavior. For this purpose we have used the parametric decomposition approach as developed by Whitt (1983), and extended by Bitran and Tirupati (1988), and Segal and Whitt (1988) for use in manufacturing networks.

One of the major questions that arises is whether the performance functions stay convex; i.e. whether we still can use marginal analysis for the allocation problems. Weber (1980) proved that in the GI/G/m case the, not explicitly known, performance functions are convex. Furthermore, Whitt (1985) developed excellent approximations for the GI/G/m case. The numerical results he obtained also show a convex behavior of the performance functions. These results indicate that marginal analysis may also provide a proper setting for the GI/G/m case.

Our experience in van Vliet and Rinnooy Kan (1989) confirms this conjecture. In this study we report on the application of the GI/G/m allocation algorithms for two real-life manufacturing networks. The results obtained show that in most cases the allocations generated by the (SA) algorithm are sufficiently close (5% relative error) to the optimal allocations. Furthermore, we are able to derive trade-off curves between the WIP and the associated minimal investment costs, and between the total number of machines in the network and the associated minimal WIP. These results indicate that the algorithms proposed in this paper indeed are useful for decisions concerning the allocation of machines in a manufacturing network.

In this paper we have discussed manufacturing problems that occur on a long-term to midterm time horizon, which usually involve strategic management decisions. On the short-term, however, management also faces important manufacturing decisions. One can think, for example, of typical product scheduling problems or lot-sizing problems. In the literature these long-term and short-term problems are most times treated separately. We feel that there is a growing need for quantitative models that capture problems both on a long-term and short-term time horizon. Some of these items are on our current research agenda.

Acknowledgements

The authors would like to thank Bill Peterson and Gerrit Timmer for their fruitful comments and suggestions on earlier versions of the manuscript.

References

Bitran, G.R., and Tirupati, D. (1987), "Trade-off curves, targeting, and balancing in manufacturing networks", University of Texas at Austin, Department of Management, Working Paper 87-08-05.

Bitran, G.R., and Tirupati, D. (1988), "Multiproduct queueing networks with deterministic routing: Decomposition approach and the notion of interference", *Management Science* 34/1, 75–100.

Buzacott, J.A., and Yao, D.D. (1986), "Flexible manufacturing systems: A review of analytical models", *Management Science* 32/7, 890–907.

Dallery, Y., and Frein, Y. (1986), "An efficient method to determine the optimal configuration of a flexible manufacturing system", in: K. Stecke and R. Suri (eds.), *Proceedings of the Second ORSA/TIMS Conference on Flexible Manufacturing Systems: Operations Research Models and Applications*.

Dyer, M.E., and Proll, L.G. (1977), "On the validity of marginal analysis for allocating servers in M/M/c queues", *Management Science* 23/9, 1019–1022.

Fox, B. (1966), "Discrete optimization via marginal analysis", *Management Science* 13/3, 210–216.

Gerla, M., and Kleinrock, L. (1977), "On the topological design of distributed computer networks", *IEEE Transactions on Communications* COM-25/1, 48–60.

Kelly, F.P. (1979), *Reversibility and Stochastic Networks*, Wiley, New York.

Kleinrock, L. (1976), *Queueing Systems Volume II: Computer Applications*, Wiley, New York.

Little, J.D.C. (1960), "A proof for the queueing formula: $L = \lambda W$", *Operations Research* 9, 383–389.

Rolfe, A.J. (1971), "A note on marginal allocation in multiple server service systems", *Management Science* 17/9, 656–658.

Segal, M., and Whitt, W. (1988), "A queueing network analyzer for manufacturing", in: *Proceedings 12th ITC*, North-Holland, Amsterdam.

Shantikumar, J.G., and Yao, D.D. (1987), "Optimal server allocation in a system of multi-server stations", *Management Science* 33/9, 1173–1180.

Tijms, H.C. (1986), *Stochastic Modelling and Analysis*, Wiley, New York.

van Vliet, M., and Rinnooy Kan, A.H.G. (1989), "Machine allocation algorithms for job shop manufacturing environments", Paper in progress.

Weber, R.W. (1980), "On the marginal benefit of adding servers to GI/I/m queues", *Management Science* 26/9, 946–951.

Weinstein, I.J., and Yu, O.S. (1973), "Comments on an integer maximization problem", *Operations Research* 21, 648–650.

Whitt, W. (1983), "The queueing network analyzer", *The Bell System Technical Journal* 62/9, 2779–2815.

Whitt, W. (1985), "Approximations for the GI/G/m queue", Working paper AT&T Bell Labs.