
The International Journal of Flexible Manufacturing Systems, 11 (1999): 353–370

Control of an Assembly System with Processing Time and Subassembly-Type Uncertainty

MATTHEW F. KEBLIS
Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor

IZAK DUENYAS
School of Business Administration, The University of Michigan, Ann Arbor

Abstract. We address the problem of controlling an assembly system in which the processing times as well as the types of subassemblies are stochastic. The quality (or performance) of the final part depends on the characteristics of the subassemblies to be assembled, which are not constant. Furthermore, the processing time of a subassembly is random. We analyze the trade-off between the increase in the potential value of parts gained by delaying the assembly operation and the inventory costs caused by this delay. We also consider the effects of processing time uncertainty. Our problem is motivated by the assembly of passive and active plates in flat panel display manufacturing. We formulate the optimal control problem as a Markov decision process. However, the optimal policy is very complex, and we therefore develop simple heuristic policies. We report the results of a simulation study that tests the performance of our heuristics. The computational results indicate that the heuristics are effective for a wide variety of cases.

Key Words: mating, electronics manufacturing, assembly, queueing control

1. Introduction

Assembly systems, consisting of several subassembly lines feeding an assembly station, are prevalent in many manufacturing environments. Two typical examples in electronics manufacturing include multiplane printed circuit boards (PCBs), which are manufactured by fabricating the layers separately then laminating them together, and flat panel displays, where "active" and "passive" layers of an electronic display are produced separately and then mated.

Previous work on assembly systems has focused on the effects of the uncertainty of the processing times at the subassembly and assembly stages (e.g., Ammar, 1980; Bhat, 1986; Bonomi, 1987; Lee and Pollock, 1989; Hopp and Simon, 1989; Duenyas and Hopp, 1992, 1993; Duenyas, 1994; Duenyas and Keblis, 1995; Rao and Suri, 1994; and Hazra and Seidmann, 1994). The focus of this body of work has been the development of approximations for the performance of different release control mechanisms for assembly systems.

Another significant source of uncertainty in some assembly systems is type uncertainty, where the type of subassembly produced by one or more of the subassembly lines or machines is uncertain. All the cited papers assume that only a single type of part is produced by each subassembly line. In contrast, in this paper, we focus on environments where there is uncertainty with respect to the type of subassembly to be produced. Furthermore, the

performance (or quality) of the final part depends on the characteristics or types of the subassemblies mated. In such an environment, the major trade-off is between the improved performance obtained by delaying the mating operation until a "match" of components of the same types is obtained and the larger inventories associated with this delay in the mating operation.

A typical example of assembly systems with processing time as well as type uncertainty is flat panel display manufacturing, where "active" and "passive" layers of an electronic display are produced by separate processes. Due to yield losses, machine failures, and the like, the time between the production of two consecutive "active" or "passive" layers is random, so that at any point in time there may be more "active" than "passive" layers or vice versa. After a passive and active layer are mated, the resulting electronic sandwich is cut into smaller pieces to produce final products (i.e., displays). The final part is defective and will have to be discarded if either the active or passive layer has a fatal defect in the location corresponding to that particular display. For this reason, each "passive" and "active" layer is inspected for fatal defects before the mating of layers and identified as being one of a set of possible types, each containing defects in specific locations. For example, when the layers will eventually be cut into 8 pieces (as is common in industrial applications), there are 2^8 types, since each of the 8 possible pieces may have zero or more fatal defects. (We note that the problem would be much simpler if the active and passive layers were first cut into smaller pieces and then only the non-defective pieces mated. However, under the currently available technology for flat panel display manufacturing, cutting the layers into smaller pieces before mating them significantly increases the defect rates. Therefore, the layers are first inspected, then mated, and then cut into smaller pieces.)

In the flat panel display environment just described, the control issues to be addressed are as follows. Given the number and types of passive and active layers already produced and not yet mated, (1) how does one decide when to release new passive or active layers into the system and (2) how does one decide which active or passive layers (if any) to mate?

A similar problem arises in the manufacturing of ball bearings, where an inner and outer race are assembled (along with a set of balls) to produce a ball bearing (Iyama, Masahiro, Goto, and Koga, 1992). Each race has a critical dimension that is a random variable, taken from a known distribution. After the races are produced, they are accurately measured at an inspection machine and classified into one of three size ranges (e.g., types). High-quality bearings are produced by mating (i.e., assembling), when possible, inner and outer races having the same size range. The "type" uncertainty is due to the variability in the dimensions of the races, and the processing time uncertainty is due to the possibility that the machines producing the inner and outer races may fail. For this problem, Iyama et al. (1992) investigate the effects of an ad-hoc mating strategy on buffers and machine blocking. They do not focus on the derivation of "optimal" mating strategies.

To our knowledge, the only paper that addresses the issue of control of assembly systems under type uncertainty is by Duenyas, Keblis, and Pollock (1997). However, Duenyas et al. (1997) assume no processing time uncertainty and that the subassembly lines produce at the same deterministic rate. Therefore, the only decision in their paper is which (if any) subassemblies to mate. Clearly, in most realistic systems, processing times are not deterministic. Furthermore, even in the rare situation where they are deterministic, it is highly unlikely that all subassembly machines (or lines) produce at the exact same rate. We

note that we cannot use the procedures developed by Duenyas et al. (1997) as heuristics for the problem where there is processing time uncertainty, because they assume that production of subassemblies occurs at deterministic time intervals and therefore impose no control over the production of more subassemblies. However, when the time of production of the next subassembly is stochastic, using the policies in Duenyas et al. (1997) will result in expected costs equal to infinity. This is because, if the subassembly lines feeding assembly are treated as independent, an assembly system is unstable even if the average production rates at all subassembly lines are the same (Harrison, 1973). For this reason, in this paper, we address the control problem when there is uncertainty in both the subassembly processing times and types.

The rest of this paper is organized as follows. Section 2 presents the problem formulation and introduces notation. In Section 3, we formulate the optimal control problem as a Markov decision process. Since the dynamic programming approach used for computing the optimal policy suffers from the "curse of dimensionality" when the number of types is greater than three, we present simple heuristics in Section 4. Section 5 presents a comparison of our heuristics against simulation for sample problems with 3 and 16 types. Section 6 concludes the paper.

2. Problem formulation and notation

We restrict ourselves to the case where two components are produced and combined to form a final product. For convenience, we refer to these as *left* and *right* halves, respectively. The act of combining halves is called *mating*. Each half is produced on a separate machine. (In this paper, we assume that each subassembly is produced on a single machine. Clearly, in many cases, subassemblies may be processed on a tandem line. We intend to analyze more complicated network structures in the future.) We assume that the processing time distribution of the machine that produces the left (right) half has mean $1/\mu_1$ ($1/\mu_2$) and variance σ_1^2 (σ_2^2). At any point in time, a machine is either on (running) or off (stopped).

When a left half is produced, it exhibits a "type" $t \in \{1, 2, \ldots, T\}$, with probability l_t; type u right halves are produced with probability r_u, where $\sum_{t=1}^{T} l_t = 1$, and $\sum_{u=1}^{T} r_u = 1$. The probability that a right (left) half is of a certain type is independent of the type of previously produced left halves and right halves. (We have found that this is a reasonable assumption in practice.) To measure the negative effect of holding inventory and long cycle times, we assume that each half kept in inventory incurs a holding cost at rate h per unit of time.

When a right half of type u is mated with a left half of type t, the resulting product has a value V_{tu}, where

$$V_{tu} > 0 \tag{1}$$

$$V_{tt} > V_{tu} \quad \text{for} \quad t \neq u \quad \text{and} \quad V_{uu} > V_{tu} \quad \text{for} \quad t \neq u \tag{2}$$

Inequality (1) implies that all matings have some value; (2) shows that mating left and right halves of the same type produces a maximum value. (For convenience, we define a *match* to be the mating of two halves of the same type.)

When there are three or more types, we will also assume the V_{tu} satisfy the additional inequality

$$V_{tt} + V_{uz} \geq V_{ut} + V_{tz} \qquad \text{for all } z \neq u, \quad u \neq t, \quad z \neq t \tag{3}$$

This inequality ensures that immediately matching two halves of the same "type" is optimal. To see this, suppose that two halves of the same type t were held in inventory without being matched. This would imply that they would eventually be mated with some other halves (e.g., with a left half of type u and a right half of type z). However, by (3) this mating would result in no greater value than matching the two halves of type t and mating the two other halves of types u and z. Since a cost is associated with holding inventory, it would be better to match the two halves of type t as soon as possible rather than hold them.

Inequalities (1) through (3) rule out some possible value matrices. However, they are reasonable for a wide variety of situations, including the problem that motivated this study. In flat panel display manufacturing, the highest value is obtained when layers that have defects at the same locations are mated. This is because once mated, a location is defective if it has defects on *either* layer and the display corresponding to that location will have to be discarded. Each layer can be represented by a vector of zeros and ones that denote whether a location, respectively, is defective or nondefective (e.g., (1, 0) represents a layer that will be divided into two pieces with the second piece defective and the first piece nondefective). Let $\boldsymbol{x}$ denote the vector associated with a passive or active layer. The unique type number $t(\boldsymbol{x})$ then can be represented as $t(\boldsymbol{x}) = \sum_{i=1}^{N} 2^i \boldsymbol{x}_i$, with N denoting the number of pieces that the layer will be cut into. Conversely, given type number t, the vector $\boldsymbol{x}(t)$ is uniquely defined to be a binary representation of t. We similarly can represent the right layer by a vector $\boldsymbol{y}$, with type $u(\boldsymbol{y})$. The value of mating or matching a left half of type t with a right half of type u then is given by $V_{tu} = r_1[\boldsymbol{x}(t) \cdot \boldsymbol{y}(u)] + r_2\{N - [\boldsymbol{x}(t) \cdot \boldsymbol{y}(u)]\}$, where r_1 is the revenue associated with a good display, r_2 is the salvage value of a defective display, and $[\boldsymbol{x}(t) \cdot \boldsymbol{y}(u)]$ is the inner product of vectors $\boldsymbol{x}(t)$ and $\boldsymbol{y}(u)$. It is straightforward to check that, when $r_1 \geq r_2$, this value function satisfies all three inequalities, (1) through (3) (Duenyas et al., 1997.)

Since an optimal policy immediately matches left and right halves of the same type, if the number of left halves of a certain type is nonzero, then the number of right halves must be zero. Therefore, we need to keep track only of the difference between the number of left halves and the right halves of a given type. The state of the system can be represented as the $\boldsymbol{T}$ vector $\{n_1, \ldots, n_T\}$, where n_t is the difference between the number of left halves of type t and right halves of type t. The fundamental problem is to determine for any given state vector $(n_1, n_2, \ldots, n_T)$, which pair (if any) of types should be mated and which machine(s) to run (left, right, or both) to maximize net value per unit time. (We note that, in the particular application that motivated this problem, the actual assembly process is much faster than the production of the subassemblies. We therefore ignore any queues at the assembly machine and focus on the inventory of parts waiting to be mated or matched.)

3. Optimal policy

In this section we give a Markov decision process (MDP) formulation for the problem described in Section 2, when processing times at both subassembly machines are assumed

to be exponential. When processing times are not exponential, one needs to keep track of the time since the last job at each machine was begun, rendering the formulation even more complicated. We therefore focus on the case where processing times are exponential. As we will show, the formulation quickly becomes intractable, even with exponential processing times. We therefore will develop a heuristic in the next section that does not assume exponential processing times.

We let

g = the minimum long-run cost per period
n = a vector that depicts the state of the system (n_t is the difference between the number of left halves of type t and the number of right halves of type t)
f_n = the relative value of being in state n
e_t = the unit vector whose tth component is 1

We use uniformization as in Lippman (1975). Without loss of generality, we also assume that $\mu_1 + \mu_2 = 1$. When this does not hold, appropriate rescaling of the problem can result in an equivalent problem with $\mu_1 + \mu_2 = 1$. The formulation consists of two cases. When all $n_t \geq 0$ for all $t = 1, \ldots, T$ or $n_t \leq 0$ for all $t = 1, \ldots, T$, then the only option is to stop one of the machines (since in this case there are no parts to mate from one of the machines). In this case, the underlying recursive equation is

$$g + f_n = \left[h\|n\| + \mu_1 \min\left\{ \sum_{t=1}^{T} l_t\left(f_{n+e_t} - V_{tt} \cdot 1_{\{n_t<0\}} \right);\ f_n \right\} + \mu_2 \min\left\{ \sum_{t=1}^{T} r_t\left(f_{n-e_t} - V_{tt} \cdot 1_{\{n_t>0\}} \right);\ f_n \right\} \right] \tag{4}$$

otherwise, we have

$$g + f_n = \min \left\{ \begin{array}{l} h\|n\| + \mu_1 \min\left\{ \sum_{t=1}^{T} l_t\left(f_{n+e_t} - V_{tt} \cdot 1_{\{n_t<0\}} \right),\ f_n \right\} \\ \quad + \mu_2 \min\left\{ \sum_{t=1}^{T} r_t\left(f_{n-e_t} - V_{tt} \cdot 1_{\{n_t>0\}} \right),\ f_n \right\} \\ \min\limits_{(u,z)\in X} \left\{ \begin{array}{l} h(\|n\| - 2) - V_{uz} + \mu_1 \min\left\{ \sum_{t=1}^{T} l_t\left(f_{n-e_u+e_z+e_t} \right. \right. \\ \quad \left. - V_{tt} \cdot 1_{\{n_t+1_{\{t=z\}}<0\}} \right),\ f_{n-e_u+e_z} \Big\} \\ \quad + \mu_2 \min\left\{ \sum_{t=1}^{T} r_t\left(f_{n-e_u+e_z+e_t} - V_{tt} \cdot 1_{\{n_t-1_{\{t=u\}}>0\}} \right),\ f_{n-e_u+e_z} \right\} \end{array} \right. \end{array} \right. \tag{5}$$

where $X = \{(u, z) : n_u > 0, n_u \cdot n_z < 0, \quad \text{and} \quad u, z = 1, \ldots, T\}$; 1_y the indicator function that equals 1 when y is true and 0 otherwise, and $\|n\| = |n_1| + |n_2| + \cdots + |n_T|$. In (4), the

term involving μ_1 pertains to the decision of either running or stopping the left machine. The term involving μ_2 pertains to the decision of either running or stopping the right machine. When a new component is produced, if it is a left half of type t, then the state becomes $n + e_t$ and a value V_{tt} is earned if there is a right half of type t in inventory. If the component produced is a right half of type t, then the state becomes $n - e_t$ and a value V_{tt} is earned if there is a left half of type t in inventory. In (5), the choice is between either not mating any halves (the top line of (5)) or mating a left half of type u with a right half of type z. The bottom line is the minimum of all possible matings of left halves of type u with right halves of type z.

Equations (4) and (5) completely define the MDP formulation for the T-type problem. Unfortunately, the optimal solution to the MDP in (4) and (5) has an extremely complicated structure. The decision to shut off one of the machines, for example, is dependent on the number *and* types of left and right halves and not only on the difference between the number of left and right halves. Moreover, as Duenyas et al. (1997) show, even for the case where there is no processing time uncertainty, the optimal policy becomes *very* complex. In fact, even if the optimal solution to (4) and (5) were available, it is questionable whether this solution could be implemented, as a huge database of the optimal decisions for each possible state would need to be stored and consulted after the production of each part. These considerations lead us to the development of two simple but effective heuristics, which we describe in the next section.

4. Heuristic solutions

The MDP formulation of the type matching problem quickly suffers from the "curse of dimensionality" as the number of types increases. For example, with four types, formulating an MDP where the number of left or right sides of a certain type is at most 40 requires over 40 million states. Therefore, for realistically sized problems (for example, with 16 types) solving the MDP is not a practical proposition. Furthermore, even if the solution were somehow available, a database that would store the optimal decision in each possible state would be hard to construct. This leads us to consider two heuristic solutions. The first heuristic (H1) is similar to one that we observed in use at the flat panel display manufacturing plant motivating this problem. This heuristic decomposes the problem into two separate problems. The first problem is when to stop the left (or right) machine, and the second problem is how to decide whether or not to mate the existing parts and which ones to mate. Heuristic 1 applies two simple thresholds to address these problems. The first threshold, a_i, is the maximum total number of left *and* right halves allowed to accumulate in inventory. If the sum of left and right halves reaches a_i, then a transportation problem is solved to determine how to mate all the available halves optimally. The second threshold, a_s, is the maximum number of left (or right) halves allowed to accumulate in inventory. If the number of left (right) halves in inventory reaches a_s, then the left (right) machine is stopped until the number of left (right) halves in inventory drops to $a_s - 1$. The best values of a_s and a_i are determined through simulation for a given problem.

H1 is implemented as follows. Every time a left (right) half is produced, it is matched with a right (left) half of the same type if any are in inventory. If no half of the same

type is available, the newly produced half is added to inventory. If the sum of the left and right halves has reached a_i, then a transportation problem is solved to decide how to mate the available halves. Where the number of left and right halves available is not equal, the solution to the transportation problem will prescribe that some of the halves go unmated and remain in inventory. The levels of left and right halves in inventory then are compared to a_s (regardless of whether a transportation problem has been solved to mate parts) to determine whether either machine should be stopped.

Despite its simplicity and ease of implementation, H1 has significant weaknesses. First of all, both thresholds have to be computed by simulation, which is time consuming. Furthermore, a new simulation study is required if any of the system parameters change. Also, to use the algorithm, the firm must have software for solving the transportation problem to decide which parts to mate (although in the computerized environment of flat panel display manufacturing, this is not a significant problem). These problems with H1 led us to develop a second heuristic, H2.

Our second heuristic (H2) also decomposes the combined problem of deciding when to shut off machines and mate parts to separate problems. However, no simulations are required to implement H2. To decide when to shut off the machines, we replace the original problem of T types with one having a single "average" type. In a problem with only one type of product, as soon as there is a single left and right half, these can be assembled. Therefore, the only decision here is on when to shut off the left or right machine.

We replace the original T-type problem with a 1-type problem with the same holding cost and machine processing times as the original problem. We assume that every time two parts are mated, a revenue of $R = (\sum_{i=1}^{T} l_i r_i V_{ii})/(\sum_{i=1}^{T} l_i r_i)$ is earned. Note that R is an average revenue computed over the maximum fraction of parts that can be matched. We therefore replace the original T-type problem with a single-type problem with the average revenue R per part. Given R, h, and the machine processing time distributions, we can compute the two thresholds a_l and a_r, by solving the following simple dynamic program:

$$g + v_i = \frac{1}{\tau}\left(2h|i| + \mu_1 \min\left\{v_{i+1} - R1_{\{i<0\}};\, v_i\right\} + \mu_2 \min\left\{v_{i-1} - R1_{\{i>0\}};\, v_i\right\}\right) \qquad (6)$$

In (6), g denotes the optimal cost per unit time, v_i denotes the relative cost of state i, and $\tau = \mu_1 + \mu_2$. With probability μ_1/τ, the next event is a completion of a left half, and the first minimization represents the choice between producing another left half (and getting a revenue R if only right halves are in inventory) or not producing another left half by shutting off the machine. The explanation for the second minimum is similar. The following theorem states the structure of the optimal policy for deciding when to shut off the machines for the described 1-type problem.

Theorem 1. *The optimal policy for the 1-type assembly control problem where the machine processing times have exponential distributions is a control-limit policy requiring only the state n (a scalar, the difference between the number of left and right halves) and two numbers* $a_l \geq 0$ *and* $a_r \leq 0$. *The optimal policy is to*

1. Stop the left machine if $i > a_l$;
2. Stop the right machine if $i < a_r$;
3. Wait for the next component produced if $a_r \le i \le a_l$.

Proof: To prove the result, it is sufficient to show that $v_i - v_{i+1}$ is increasing in i. To see this, note that, if it is optimal to stop the left machine for some $i > 0$, then $v_i \ge v_{i+1} - R$. Since $v_{i+1} - v_{i+2} \ge v_i - v_{i+1}$, combining these two equations, we have that $v_{i+1} \ge v_{i+2} - R$ as well, which proves the existence of the threshold a_l. The argument for a_r is similar. The fact that $v_i - v_{i+1}$ is increasing in i can be shown by induction. Consider a value iteration algorithm for obtaining the v_i values, where

$$v_i^{k+1} = \frac{1}{\tau}\left(2h|i| + \mu_1 \min\left\{v_{i+1}^k - R1_{\{i<0\}};\, v_i^k\right\} + \mu_2 \min\left\{v_{i-1}^k - R1_{\{i>0\}};\, v_i^k\right\}\right) \quad (7)$$

where v_k^i denotes the relative value of being in state i at the kth iteration of the value iteration algorithm. Let $v_i^0 = 0$ for all i. Then, clearly, $v_i^0 - v_{i+1}^0$ is increasing in i. It is straightforward to show that, if $v_i^k - v_{i+1}^k$ is increasing in i, then $v_i^{k+1} - v_{i+1}^{k+1}$ also is increasing in i. Since $v_i^k - v_{i+1}^k \to v_i - v_{i+1}$ and the desired property is preserved at every iteration, the result is shown. □

Once the threshold value for shutting off either the left or the right machine is obtained, our decomposition heuristic also computes threshold values a_{tu} for mating a left half of type t with a right half of type u, for all $t = 1, \ldots, T$ and $u = 1, \ldots, T$, $u \ne t$. To do this, we decompose the original T-type problem into simpler 2-type problems. We solve 2-type problems (ignoring the existence of the other types) to obtain the thresholds for mating those two types. For this, the production probabilities first have to be conditioned on producing only those particular types t and u,

$$l_t' \equiv \Pr\{\text{left type } t \text{ produced} \mid \text{only left } t \text{ or } u \text{ produced}\} = \frac{l_t}{l_t + l_u} \equiv 1 - l_u' \quad (8)$$

$$r_t' \equiv \Pr\{\text{right type } t \text{ produced} \mid \text{only right } t \text{ or } u \text{ produced}\} = \frac{r_t}{r_t + r_u} \equiv 1 - r_u' \quad (9)$$

We also need to rescale the holding costs. In an actual 2-type mating problem, the expected time until a new left (right) half arrives is $1/\mu_1 (1/\mu_2)$, when the left (right) machine is not stopped. With more than two types, the expected time until the arrival of another right half of type t or u equals $1/\mu_2(r_t + r_u)$. Thus, each left half of type t or u in inventory incurs an expected holding cost of $h/\mu_2(r_t + r_u)$ until the arrival of the next right half of either type. Similarly, each right half of type t or u incurs an expected cost $h/\mu_1(l_t + l_u)$ until the arrival of the next left half of either type. Therefore, when we rescale the probabilities for types t and u, we also need to rescale the holding cost values. We assign the rescaled holding cost value $h' \equiv [h/2(l_t + l_u)] + [h/2(r_t + r_u)]$ to be the average of these two values.

Given the conditional probabilities, l_t', r_t', l_u', r_u', the rescaled holding cost h', the values μ_1, μ_2, V_{tu}, V_{ut}, V_{tt}, and V_{uu} from the original problem and the thresholds for shutting off

the left and right machines a_l and a_r, we can compute the optimal thresholds a_{tu} and a_{ut} for mating a half of t and a half of u. To do this, note that, given the values of a_{tu} and a_{ut}, the system can be modeled as a simple Markov process, for which we can compute the average cost per unit time. In this Markov process, the state of the system is (i, j), where i denotes the difference between the number of left and the number of right halves of type t and j denotes the difference between the number of left and right halves of type u. Except at the boundaries (defined by a_l, a_r, a_{tu}, and a_{ut}), the possible transitions from (i, j) are to states

1. $(i+1, j)$ with rate $\mu_1 l'_t$ corresponding to the arrival of a left half of type t;
2. $(i, j+1)$ with rate $\mu_1 l'_u$ corresponding to the arrival of a left half of type u;
3. $(i-1, j)$ with rate $\mu_2 r'_t$ corresponding to the arrival of a right half of type t;
4. $(i, j-1)$ with rate $\mu_2 r'_u$ corresponding to the arrival of a right half of type u.

Similarly, except at the boundaries, in state (i, j) costs are incurred with rate

$$h(|i| + |j|) - \mu_2\left(r'_t V_{tt} 1_{\{i>0\}} + r'_u V_{uu} 1_{\{j>0\}}\right) - \mu_1\left(l'_t V_{tt} 1_{\{i<0\}} + l'_u V_{uu} 1_{\{j<0\}}\right)$$

The steady state probabilities as well as the average cost per unit time for this Markov process can be computed easily and the optimal values of a_{tu} and a_{ut} can be found by complete enumeration.

Once the thresholds a_l and a_r for shutting off the left and right machines and the thresholds for mating, a_{tu} and a_{ut} for all $t = 1, \ldots, T$ and $u = 1, \ldots, T$ have been computed, the implementation of H2 is straightforward. Every time a new left (right) half is produced, it is mated with a right (left) half of the same type if there are any in inventory. If no half is available, then the newly produced half is added to inventory. For any $t = 1, \ldots, T$ and $u = 1, \ldots, T$, if there are at least a_{tu} left halves of type t and right halves of type u, then a left side of type t is mated with a right side of type u. After carrying out any matching or mating, the number of left and right halves in inventory are compared to a_l and a_r, respectively, to determine whether either machine should be stopped.

Figure 1 demonstrates the main difference between the heuristic H2 (described by the bolder lines) and the structure of the optimal solution in a 2-type problem. In the area where $n_1 > 0$, $n_2 < 0$, or $n_1 > 0, n_2 < 0$ (areas 2 and 4 in figure 1), the heuristic results in simple rectangular regions that describe the mating of types 1 and 2 (whereas the optimal policy has a more complicated elliptical shape); similarly in regions 1 and 3, the heuristic results in simple triangular regions for shutting the machines (whereas the optimal policy again has a more complicated elliptical shape). However, as we show next, despite the differences in shape between the heuristic and optimal policies, H2 performs very well.

H2 is at least as simple to implement in practice as H1. The thresholds can be computed very quickly. To obtain the thresholds for stopping the machines, one needs to solve a single MDP with state space size $2M$. (M needs to be chosen large enough that it is higher than the values at which the MDP would decide to shut off the machines. An appropriate upper bound for M is $M = R/(h * (\mu_1 + \mu_2))$, since if the number of jobs waiting for mating is higher than that, the amount of holding cost incurred until the arrival of the next part is higher than the revenue of a part.) Similarly, to obtain the threshold values for mating parts, one would need to solve $T * (T-1)$ MDPs, each with state space size $4M^2$. For problems

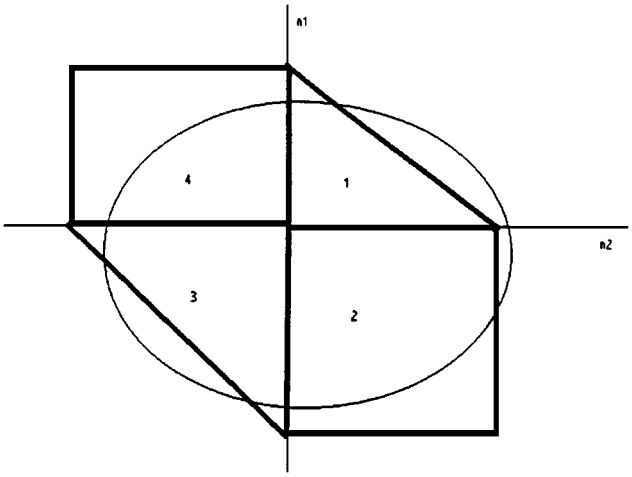

Figure 1. The structure of the optimal policy and H2 for a 2-type problem.

with as many as 16 types, (typical in the flat panel display environment), the computation of *all* the thresholds (which have to be computed only once) takes less than an hour of CPU time on a Sun Sparcstation 20. Checking to see if any threshold has been exceeded also is simple, especially in many computerized electronics manufacturing environments. Furthermore, unlike in H1, the heuristic requires no simulation experiments or access to transportation problem solvers on the plant floor.

Finally, we note that, although we described both heuristics for the case where the processing times are assumed to be exponential, they easily can be adapted to nonexponential processing times. In particular, as the optimal thresholds for H1 are computed by simulation, whether the processing times are exponential or not makes no difference. In the case of H2, approximating the processing times by appropriate phase-type distributions results in Markov chains for which the optimal thresholds once again can be computed, albeit at the cost of larger state spaces.

From an implementation point of view, H1 and H2 offer interesting contrasts. H1 requires significant work up front to set the threshold values compared to the work required to set the threshold values for H2 (unless the number of types is very high, in which case H2 also requires a very significant amount of work; as can be seen, the number of MDPs one needs to solve for H2 is quadratic to the number of parts). H2 requires more significant monitoring than H1, as the amount of inventory of each type has to be monitored and compared to the threshold levels at all times. Which heuristic gets implemented in practice will depend on the relative costs of monitoring inventory at all times.

Finally, we note that, in many practical situations, some cost is incurred each time a machine is turned on or off. This cost might reflect the cost of raw material lost during the initial period in which the machine is adjusted or the operator time spent monitoring the machine. The formulation for the optimal policy (4) and (5) can be easily changed to include a cost S whenever a machine is turned on. In this case, the state space is represented by a vector (n, i) where $\boldsymbol{n}$ is a T-vector denoting, as before, the difference between the number

of left halves and right halves of type t for $t = 1, \ldots, T$ and $i = 0, 1, 2$ denotes the state of the machines. In this case, $i = 0$ would represent the case where both machines are running, $i = 1$ is the case where only the left machine is running, and $i = 2$ is the case where only the right machine is running. (It can never be optimal to have both machines off.)

A variation of H2 for the case where costs are incurred whenever machines are turned on, H2S, can be constructed in the following manner. First, we replace the original T-type problem by a 1-type problem as before and find, for each state, which machines should be on or off by solving the following variation of (6):

$$g + v_{i,0} = \frac{1}{\tau}\left(2h|i| + \mu_1 \min\left\{v_{i+1,0} - R1_{\{i<0\}};\ v_{i,2}\right\} + \mu_2 \min\left\{v_{i-1,0} - R1_{\{i>0\}};\ v_{i,1}\right\}\right)$$

$$g + v_{i,1} = \min\begin{cases} S + \frac{1}{\tau}\left(2h|i| + \mu_1 \min\left\{v_{i+1,0} - R1_{\{i<0\}};\ v_{i,2}\right\}\right. \\ \qquad \left. + \mu_2 \min\left\{v_{i-1,0} - R1_{\{i>0\}};\ v_{i,1}\right\}\right) \\ \frac{1}{\tau}\left(2h|i| + \mu_1\left\{v_{i+1,1} - R1_{\{i<0\}}\right\} + \mu_2 v_{i,1}\right) \end{cases}$$

$$g + v_{i,2} = \min\begin{cases} S + \frac{1}{\tau}\left(2h|i| + \mu_1 \min\left\{v_{i+1,0} - R1_{\{i<0\}};\ v_{i,2}\right\}\right. \\ \qquad \left. + \mu_2 \min\left\{v_{i-1,0} - R1_{\{i>0\}};\ v_{i,1}\right\}\right) \\ \frac{1}{\tau}\left(2h|i| + \mu_1 v_{i,2} + \mu_2\left\{v_{i-1,2} - R1_{\{i>0\}}\right\}\right) \end{cases} \tag{10}$$

The solution to (10) will give the states (in terms of difference of the total number of left halves and right halves) in which either both machines or only the right or only the left machine should be on. The second step of H2 (i.e., decomposition for obtaining the thresholds for mating) remains identical for H2S. In the next section, we test the performance of H1, H2, and H2S.

5. Computational results

We conducted a simulation study to test the performance of the heuristics. Since problems with more than three types become extremely difficult to solve optimally (as noted, an MDP formulation of a problem with four types can easily require over 40 million states), the heuristics were tested against the optimal policy only for problems with two and three types. We also compared the heuristics H1 and H2 for typical 16-type problems arising in flat panel display manufacturing by using simulation.

Tables 1 through 3 show how the two heuristics performed on 36 test problems with three types and no setup costs. The results in Table 1 are for systems where both machines have exponential processing time distributions with rate 0.50. In Table 2, the results are for systems where each left machine has an exponential processing time distribution with rate 0.60 and each right machine has an exponential processing time distribution with rate 0.40. In Table 3, the results are for systems where each left machine has an exponential processing time distribution with rate 0.66 and each right machine has an exponential processing time distribution with rate 0.33. For each use of H1, simulation was used to compute the profit-maximizing threshold values. Tables 1 through 3 show the average profit per unit time

Table 1. Results for 3-type cases, $\mu_1 = 0.50$, $\mu_2 = 0.50$.

Case	V_{11}, V_{12}, V_{13} V_{21}, V_{22}, V_{23} V_{31}, V_{32}, V_{33}	l_1, l_2, l_3 r_1, r_2, r_3	h	H1 (% suboptimal)	H2 (% suboptimal)	= Optimal
1	10, 7, 4	0.33, 0.33, 0.33				
	7, 10, 7	0.33, 0.33, 0.33	0.05	4.21 (0.9)	4.23 (0.5)	4.25
	4, 7, 10					
2	10, 7, 4	0.33, 0.33, 0.33				
	7, 10, 7	0.33, 0.33, 0.33	0.02	4.52 (0.0)	4.52 (0.0)	4.52
	4, 7, 10					
3	10, 7, 4	0.50, 0.30, 0.20				
	7, 10, 7	0.50, 0.30, 0.20	0.05	4.24 (0.5)	4.25 (0.2)	4.26
	4, 7, 10					
4	10, 7, 4	0.50, 0.30, 0.20				
	7, 10, 7	0.50, 0.30, 0.20	0.02	4.52 (0.2)	4.52 (0.2)	4.53
	4, 7, 10					
5	10, 7, 4	0.60, 0.20, 0.20				
	7, 10, 7	0.20, 0.20, 0.60	0.05	3.32 (1.2)	3.36 (0.0)	3.36
	4, 7, 10					
6	10, 7, 4	0.60, 0.20, 0.20				
	7, 10, 7	0.20, 0.20, 0.60	0.02	3.49 (0.9)	3.52 (0.0)	3.52
	4, 7, 10					
7	10, 6, 2	0.33, 0.33, 0.33				
	6, 6, 2	0.33, 0.33, 0.33	0.05	2.36 (2.1)	2.38 (1.2)	2.41
	2, 2, 2					
8	10, 6, 2	0.33, 0.33, 0.33				
	6, 6, 2	0.33, 0.33, 0.33	0.02	2.61 (0.4)	2.62 (0.0)	2.62
	2, 2, 2					
9	10, 6, 2	0.50, 0.30, 0.20				
	6, 6, 2	0.50, 0.30, 0.20	0.05	2.94 (1.7)	2.98 (0.3)	2.99
	2, 2, 2					
10	10, 6, 2	0.50, 0.30, 0.20				
	6, 6, 2	0.50, 0.30, 0.20	0.02	3.18 (0.9)	3.20 (0.3)	3.21
	2, 2, 2					
11	10, 6, 2	0.60, 0.20, 0.20				
	6, 6, 2	0.20, 0.20, 0.60	0.05	1.82 (2.7)	1.86 (0.5)	1.87
	2, 2, 2					
12	10, 6, 2	0.60, 0.20, 0.20				
	6, 6, 2	0.20, 0.20, 0.60	0.02	1.96 (1.5)	1.99 (0.0)	1.99
	2, 2, 2					

Table 2. Results for 3-type cases, $\mu_1 = 0.60$, $\mu_2 = 0.40$.

Case	V_{11}, V_{12}, V_{13} V_{21}, V_{22}, V_{23} V_{31}, V_{32}, V_{33}	l_1, l_2, l_3 r_1, r_2, r_3	h	H1 (% suboptimal)	H2 (% suboptimal)	= Optimal
13	10, 7, 4 7, 10, 7 4, 7, 10	0.33, 0.33, 0.33 0.33, 0.33, 0.33	0.05	3.51 (2.8)	3.51 (2.8)	3.61
14	10, 7, 4 7, 10, 7 4, 7, 10	0.33, 0.33, 0.33 0.33, 0.33, 0.33	0.02	3.72 (1.1)	3.70 (1.6)	3.76
15	10, 7, 4 7, 10, 7 4, 7, 10	0.50, 0.30, 0.20 0.50, 0.30, 0.20	0.05	3.52 (3.0)	3.53 (2.8)	3.63
16	10, 7, 4 7, 10, 7 4, 7, 10	0.50, 0.30, 0.20 0.50, 0.30, 0.20	0.02	3.70 (1.9)	3.70 (1.9)	3.77
17	10, 7, 4 7, 10, 7 4, 7, 10	0.60, 0.20, 0.20 0.20, 0.20, 0.60	0.05	2.84 (3.7)	2.93 (0.7)	2.95
18	10, 7, 4 7, 10, 7 4, 7, 10	0.60, 0.20, 0.20 0.20, 0.20, 0.60	0.02	2.95 (1.7)	3.00 (0.0)	3.00
19	10, 6, 2 6, 6, 2 2, 2, 2	0.33, 0.33, 0.33 0.33, 0.33, 0.33	0.05	1.96 (5.8)	2.00 (3.8)	2.08
20	10, 6, 2 6, 6, 2 2, 2, 2	0.33, 0.33, 0.33 0.33, 0.33, 0.33	0.02	2.13 (3.2)	2.15 (2.3)	2.20
21	10, 6, 2 6, 6, 2 2, 2, 2	0.50, 0.30, 0.20 0.50, 0.30, 0.20	0.05	2.47 (3.9)	2.49 (3.1)	2.57
22	10, 6, 2 6, 6, 2 2, 2, 2	0.50, 0.30, 0.20 0.50, 0.30, 0.20	0.02	2.63 (2.2)	2.64 (1.9)	2.69
23	10, 6, 2 6, 6, 2 2, 2, 2	0.60, 0.20, 0.20 0.20, 0.20, 0.60	0.05	1.55 (7.2)	1.64 (1.8)	1.67
24	10, 6, 2 6, 6, 2 2, 2, 2	0.60, 0.20, 0.20 0.20, 0.20, 0.60	0.02	1.66 (3.5)	1.71 (0.6)	1.72

Table 3. Results for 3-type cases, $\mu_1 = 0.66,\ \mu_2 = 0.33$.

Case	V_{11}, V_{12}, V_{13} V_{21}, V_{22}, V_{23} V_{31}, V_{32}, V_{33}	l_1, l_2, l_3 r_1, r_2, r_3	h	H1 (% suboptimal)	H2 (% suboptimal)	= Optimal
25	10, 7, 4 7, 10, 7 4, 7, 10	0.33, 0.33, 0.33 0.33, 0.33, 0.33	0.05	2.87 (4.0)	2.92 (2.3)	2.99
26	10, 7, 4 7, 10, 7 4, 7, 10	0.33, 0.33, 0.33 0.33, 0.33, 0.33	0.02	3.05 (1.9)	3.07 (1.3)	3.11
27	10, 7, 4 7, 10, 7 4, 7, 10	0.50, 0.30, 0.20 0.50, 0.30, 0.20	0.05	2.89 (4.0)	2.93 (2.7)	3.01
28	10, 7, 4 7, 10, 7 4, 7, 10	0.50, 0.30, 0.20 0.50, 0.30, 0.20	0.02	3.06 (2.2)	3.07 (1.9)	3.13
29	10, 7, 4 7, 10, 7 4, 7, 10	0.60, 0.20, 0.20 0.20, 0.20, 0.60	0.05	2.38 (3.6)	2.46 (0.4)	2.47
30	10, 7, 4 7, 10, 7 4, 7, 10	0.60, 0.20, 0.20 0.20, 0.20, 0.60	0.02	2.46 (2.0)	2.50 (0.4)	2.51
31	10, 6, 2 6, 6, 2 2, 2, 2	0.33, 0.33, 0.33 0.33, 0.33, 0.33	0.05	1.62 (5.8)	1.66 (3.5)	1.72
32	10, 6, 2 6, 6, 2 2, 2, 2	0.33, 0.33, 0.33 0.33, 0.33, 0.33	0.02	1.75 (3.8)	1.78 (2.2)	1.82
33	10, 6, 2 6, 6, 2 2, 2, 2	0.50, 0.30, 0.20 0.50, 0.30, 0.20	0.05	2.03 (4.7)	2.08 (2.3)	2.13
34	10, 6, 2 6, 6, 2 2, 2, 2	0.50, 0.30, 0.20 0.50, 0.30, 0.20	0.02	2.18 (2.2)	2.19 (1.8)	2.23
35	10, 6, 2 6, 6, 2 2, 2, 2	0.60, 0.20, 0.20 0.20, 0.20, 0.60	0.05	1.30 (7.8)	1.38 (2.1)	1.41
36	10, 6, 2 6, 6, 2 2, 2, 2	0.60, 0.20, 0.20 0.20, 0.20, 0.60	0.02	1.38 (4.2)	1.43 (0.7)	1.44

obtained by H1, H2 and the optimal policy (obtained by solving the complete MDP formulation), as well as the percentage suboptimality of H1 and H2. The profits achieved by H1 and H2 were computed by simulation.

Cases 1–36 were selected to cover a wide range of operating conditions. Cases 1–6, 13–18, and 25–30 represent situations where the value of the mated part gets lower as the "difference" between the halves increases. (For example, different types may correspond to different tolerances, and the halves with the same tolerances may have the best fit.) Cases 7–12, 19–24, and 31–36 represent situations where different "qualities" are associated with each type and an assembled component is worth only as much as its lowest-quality part.

Tables 1 through 3 show that both heuristics perform well for all 36 cases. H2 performs as well as or better than H1 in all but 1 of the 36 cases. In fact, the average suboptimality of H2 was 1.4% over the 36 test cases. This is encouraging, since it is much easier to compute the parameters of H2; the thresholds required to implement H2 were calculated in several minutes for each of the examples. The simulation runs required to determine the thresholds for H1, by comparison, took as long as several hours of CPU time on a Sun Sparcstation 20. The performance of the heuristics declines slightly as the processing rates of the left and right machines become more unequal. We also note that H2 performed better in the examples where one of the types had a large probability, as in example 36. In this case, the probability of a type-1 left type is 0.6, as is the probability of a right type of type 3. Clearly, the most important decision here is the threshold for mating left type ones with right type threes. It is very important that the heuristic get this one threshold right, and its performance in estimating the other thresholds is not as important as the heuristic does not face the other decisions as often. Another interesting observation is that the heuristic tends to do slightly better at the lower holding cost value of 0.02. We believe that the reason for this is that, when holding costs are lower, the thresholds are higher. Under- or overestimating a high threshold value by one has a much smaller influence on performance than under- or overestimating a low threshold value.

Tables 4 and 5 show how the heuristics H1 and H2S performed on 12 test problems with two types and setup costs of 0.20 and 2.00. The results in Table 4 are for systems where both machines have exponential processing time distributions with rate 0.50. In Table 5, the results are for systems where each left machine has an exponential processing time distribution with rate 0.66 and each right machine has an exponential processing time distribution with rate 0.33. For each use of H1, simulation was used to compute the profit maximizing threshold values. Tables 4 and 5 show the average profit per unit time obtained by H1, H2S, and the optimal policy (obtained by solving the complete MDP formulation) as well as the percentage suboptimality of H1 and H2S. The profits achieved by H1 and H2S were computed by simulation. In Cases 37, 39, 41, 43, 45, and 47 the setup cost is 2.00. In Cases 38, 40, 42, 44, 46, and 48 the setup cost is 0.20.

Tables 4 and 5 show that H2S performs better than H1 in all the cases. The average suboptimality of H2S was 3.1% over the 12 test cases. The tables show that both heuristics perform better when the machine processing rates are equal (which indicates that the heuristic is having greater problems estimating when to shut off and turn back on machines in the case with setup costs and unequal processing rates).

Table 4. Results for 2-type cases with setup costs $\mu_1 = \mu_2 = 0.50$.

Case	V_{11}, V_{12} V_{21}, V_{22}	l_1, l_2, l_3 r_1, r_2, r_3	h	H2S (% suboptimal)	H1 (% suboptimal)	Optimal
37	10, 7	0.5, 0.5				
	7, 10	0.5, 0.5	0.02	4.59 (0.0)	4.52 (1.5)	4.59
38	10, 7	0.5, 0.5				
	7, 10	0.5, 0.5	0.02	4.60 (0.0)	4.57 (0.7)	4.60
39	10, 6	0.5, 0.5				
	6, 6	0.5, 0.5	0.02	3.64 (0.0)	3.58 (1.6)	3.64
40	10, 6	0.5, 0.5				
	6, 6	0.5, 0.5	0.02	3.65 (0.0)	3.64 (0.3)	3.65
41	10, 7	0.67, 0.33				
	7, 10	0.67, 0.33	0.02	4.59 (0.0)	4.56 (0.6)	4.59
42	10, 7	0.67, 0.33				
	7, 10	0.67, 0.33	0.02	4.60 (0.0)	4.60 (0.0)	4.60

Table 5. Results for 2-type cases with setup costs $\mu_1 = 0.66,\ \mu_2 = 0.33$.

Case	V_{11}, V_{12} V_{21}, V_{22}	l_1, l_2, l_3 r_1, r_2, r_3	h	H2S (% suboptimal)	H1 (% suboptimal)	Optimal
43	10, 7	0.5, 0.5				
	7, 10	0.5, 0.5	0.02	2.82 (9.6)	2.74 (12.2)	3.12
44	10, 7	0.5, 0.5				
	7, 10	0.5, 0.5	0.02	3.07 (2.5)	2.74 (13.0)	3.15
45	10, 6	0.5, 0.5				
	6, 6	0.5, 0.5	0.02	2.20 (11.3)	2.08 (16.1)	2.48
46	10, 6	0.5, 0.5				
	6, 6	0.5, 0.5	0.02	2.44 (3.2)	2.10 (16.7)	2.52
47	10, 7	0.67, 0.33				
	7, 10	0.67, 0.33	0.02	2.84 (8.9)	2.76 (11.5)	3.12
48	10, 7	0.67, 0.33				
	7, 10	0.67, 0.33	0.02	3.09 (2.2)	2.77 (12.3)	3.16

We also compared the performance of the two heuristics on a typical problem from flat panel display manufacturing. We used simulation to compare the performance of the heuristics in this case. As described in Section 1, in flat panel display manufacturing, "active" and "passive" halves are inspected to determine the location of the defects. The plates are then assembled and cut into smaller products. A typical example has the "passive" and "active" halves cut into four pieces after being mated to produce four displays. In this case,

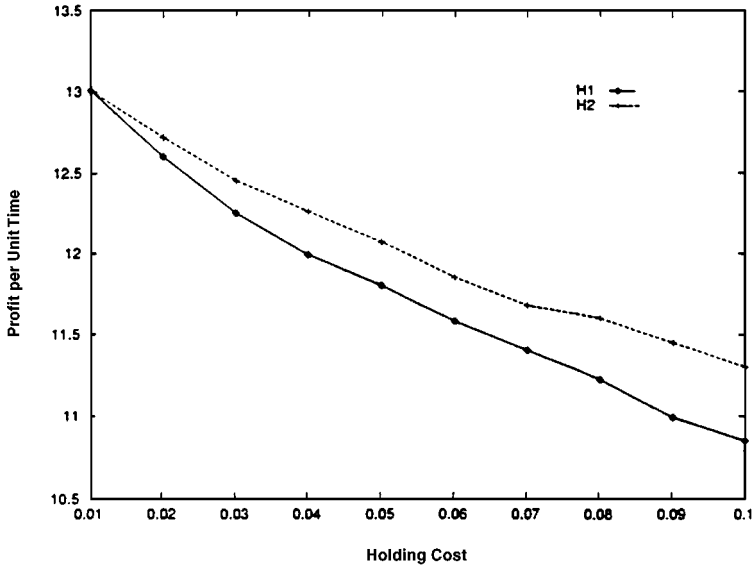

Figure 2. Performance of H1 and H2 for 16-type problem.

since each half could have defects in four different locations, there are 16 possible types of halves. Suppose that the probability of defect in any location is 0.3, independent of defects at other locations, and a display is worth \$10 if it has no defects on either half. Therefore, mating "passive" and "active" halves that both have no defects is worth \$40. On the other hand, mating an active plate that has a defect in the location of the third display with a passive plate that has defects in the location of the second and fourth displays will result in production of only one 1 display (in the location of the first display) and the total mate is worth only \$10. We assumed that the machines producing the left and right halves had exponential distributions with rate 0.5.

Since this problem has 16 types, it cannot be solved to optimality using dynamic programming. Even if the maximum inventory of a given type were limited to at most 40 units, there would be over 10^{28} states in the MDP. Figure 2 shows the performance of H2 and H1 as a function of h. For all the holding cost values displayed, H2 outperforms H1, although in this case the difference between the two heuristics is smaller than in the three-machine cases.

6. Conclusion and further research

In this paper, we formulated an assembly control problem that arises in many manufacturing environments, including flat panel display manufacturing, and presented several heuristics. Although the heuristics performed well on the examples without setup costs, further research is needed on the problem with setup costs. Further research also should focus on more complicated systems than the one considered here (e.g., subassembly lines instead of machines).

Acknowledgments

This work was supported in part by Grants Nos: DMI-9308290 and DMI-9424596 from the National Science Foundation and a grant from the Center for Display Technology and Manufacturing at the University of Michigan.

References

Ammar, M.H., "Modelling and Analysis of Unreliable Manufacturing Assembly Networks with Finite Storage," MIT Laboratory for Information and Decision Sciences, Report No. LIDS-TH-1004 (1980).

Bhat, U.N., "Finite Capacity Assembly-Like Queues," *Queueing Systems, Theory and Applications*, Vol. 1, No. 1, pp. 85–102 (1986).

Bonomi, F., "An Approximate Analysis for a Class of Assembly-Like Queues," *Queueing Systems: Theory and Applications*, Vol. 1, No. 2, pp. 289–300 (1987).

Duenyas, I., "Estimating the Throughput of a Cyclic Assembly System," *International Journal of Production Research*, Vol. 32, No. 7, pp. 1403–1419 (1994).

Duenyas, I. and Hopp, W.J., "CONWIP Assembly with Deterministic Processing and Random Outages," *IIE Transactions*, Vol. 24, No. 1, pp. 97–111 (1992).

Duenyas, I. and Hopp, W.J., "Estimating the Throughput of an Exponential CONWIP Assembly System," *Queueing Systems: Theory and Applications*, Vol. 14, No. 1, pp. 135–157 (1993).

Duenyas, I. and Keblis, M.F., "Release Policies for Assembly Systems," *IIE Transactions*, Vol. 27, No. 4, pp. 507–518 (1995).

Duenyas, I., Keblis, M.F., and Pollock, S.M., "Dynamic Type Mating," *Management Science*, Vol. 43, No. 6, pp. 751–763 (1997).

Gershwin, S.B., "Assembly/Disassembly Systems: An Efficient Decomposition Algorithm for Tree Structured Networks," *IIE Transactions*, Vol. 23, No. 2, pp. 302–314 (1991).

Harrison, J.M., "Assembly-Like Queues," *Journal of Applied Probability*, Vol. 10, No. 2, pp. 354–374 (1973).

Hazra, J. and Seidmann, A., "Performance Evaluation of Closed Tree-Structured Assembly Systems," Technical Report OP93-01, William E. Simon Graduate School of Business Administration, University of Rochester, Rochester, NY (1994).

Hernandez-Lerma, O. *Adaptive Markov Control Processes*, Springer-Verlag, New York (1989).

Hopp, W.J. and Simon, J.T., "Bounds and Heuristics for Assembly-Like Queues," *Queueing Systems: Theory and Applications*, Vol. 4, No. 1, pp. 137–147 (1989).

Iyama, T., Masahiro, M., Goto, S., and Koga, T., "A Race Matching Method for Ball Bearing Manufacture," Department of Mechanical Engineering, Iwate University, Morioka, Japan (1992).

Lee, H.S. and Pollock, S.M., "Approximate Analysis for the Merge Configuration of an Open Queueing Network with Blocking," *IIE Transactions*, Vol. 21, No. 1, pp. 122–129 (1989).

Lippman, S.A., "Applying a New Device in the Optimization of Exponential Queueing Systems," *Operations Research*, Vol. 23, No. 3, pp. 687–710 (1975).

Lipper, E.H. and Sengupta, B., "Assembly-Like Queues with Finite Capacity: Bounds, Asymptotics and Approximations," *Queueing Systems: Theory and Applications*, Vol. 1, No. 1, pp. 67–84 (1986).

Liu, Y.C. and Perros, H.G., "A Decomposition Procedure for the Analysis of a Closed Fork/Join Queueing System," *IEEE Transactions on Computers*, Vol. C-40, No. 2, pp. 365–370 (1991).

Rao, P.C. and Suri, R., "Approximate Queueing Network Models for Closed Fabrication/Assembly Systems. Part 1: Single Level Systems," *Production and Operations Management*, Vol. 3, No. 4, pp. 244–275 (1994).

Saboo, S. and Wilhelm, W.E., "An Approach for Modeling Small-Lot Assembly Networks," *IIE Transactions*, Vol. 18, No. 3, pp. 322–334 (1986).