IIE Transactions (1998) **30**, 31–39

Base-stock control for single-product tandem make-to-stock systems

IZAK DUENYAS and PRAYOON PATANA-ANAKE

Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI, 48109, USA

Received October 1995 and accepted December 1996

In this paper, we consider a multiple-stage tandem production/inventory system producing a single product. Processing time at each stage is assumed to have a general stationary processing time distribution. The cost of holding work-in-process (WIP) inventory is different at each stage. Therefore, decisions on when to release work to the system as well as when to transfer WIP from one stage to another need to be made. We formulate this problem of release/production control as a Markov decision process. However, the optimal policy is rather complex, making its implementation impracticable in practice. We therefore investigate the performance of simple base stock policies. Our approach aggregates several stages into one and uses a simple approximation to compute 'approximately optimal' base stock levels. We present the results of a simulation study that tests the performance of our approximation in estimating the best base stock levels, and the performance of base stock policies as compared with the optimal policy.

1. Introduction

Recently, considerable attention has been devoted to developing effective control mechanisms for production/inventory systems. Driven in part by the success of Japanese 'pull production systems', researchers have focused on the analysis of mechanisms that dictate when work will be released to a production system, as well as the conditions under which work can be transferred from one stage to another in the production process.

Most research so far has focused on the performance evaluation of specific policies. The performance of the kanban control mechanism has been studied for tandem make-to-order systems (see [1,2]), tandem make-to-stock systems [3], as well as for assembly systems [4]. Chang and Yih [5,6] develop a generic kanban system for dynamic environments. Askin *et al.* [7] and Mitsawi and Askin [8] address the problem of determining the number of kanbans in multi-item just-in-time systems and develop effective production planning policies for a multi-item, single-stage kanban system. Research on the performance of the CONWIP release mechanism [9] has resulted in approximations for the throughput of tandem systems [10], and the variance of the output process [11]. Duenyas and Hopp [12] and Duenyas [13] have also derived approximations for the throughput of assembly systems under the CONWIP release mechanism. Buzacott *et al.* [14], Buzacott and Shanthikumar [15] and Lee and Zipkin [16] have developed approximations for the performance of base-stock policies and compare their approximations with simulation for systems with two and three machines. Rubio and Wein [17] extended the CONWIP system to the make-to-stock case. Under their policy, a new unit of product is released to the shop floor whenever the total WIP plus finished goods inventory (where backordered demand represents negative inventory) falls below a specified base stock level. They show how the optimal inventory level can be analytically computed, under product-form assumptions. Uzsoy *et al.* [18] provide a detailed survey of release control mechanisms in the context of the semiconductor industry.

The performance of different control rules has very rarely been compared. Muckstadt and Tayur [2] and Duenyas and Keblis [4] compare the performance of kanban and CONWIP. The purpose of the comparisons is to find out which policy achieves a target throughput level with the minimum possible WIP (equivalently, which policy achieves a higher throughput for a given WIP level). This objective implicitly assumes that WIP costs are the same at each stage of production. However, in a manufacturing system with many stages of production, the cost of holding a unit of WIP is not likely to be the same throughout the production process. This is because value is added to the product at each stage of the production process in the form of labor hours spent processing the product, and materials used at the different stages. Even though the value added at any one individual machine may be small, the difference between the value of a unit of WIP at the last stage of production and at the first stage of production is significant in most manufacturing systems. In some cases, production of a product requires work at several different plants and a significant part of the value added is the transportation costs of transferring the parts from one plant to another. A modelling approach that penalizes holding inventory more severely at each stage of production is required to handle such situations. Clearly, it is not necessary to compute the value added after each minor operation of the production process. This would be unnecessarily complex, especially in an environment with thousands of operations, and computing the optimal parameters for any policy for a system modelled in such detail is unlikely to be tractable. Therefore, we consider an approach that models several stages, each of which consists of multiple operations with a distinct cost of inventory for each stage. This cost can be taken to be the average cost of inventory at that stage. This modelling approach enables us to find approximately optimal base stock levels for large systems consisting of many machines very rapidly.

In a recent paper, Veatch and Wein [19] considered the optimal control of a two-stage make-to-stock system where each stage consists of a single exponential machine. They derived sufficient conditions under which it would be optimal to hold no finished goods or WIP inventory. They also used simulation to compare the performance of base stock, kanban, and fixed buffer policies against the optimal policy computed by using dynamic programming. In their simulation experiments, the base-stock policy performed very well except when the upstream machine is slow. The base-stock policy that they analyze releases a new job to the first machine in the system immediately whenever a finished good is demanded. As they explain, this results in unnecessary stockpiling of WIP when there are many backorders. To prevent this phenomenon from occurring in this paper, we focus on a base-stock policy with a limit on the WIP on the shop floor. This is motivated by the observation that if a system has enough WIP to keep the bottleneck starvation probability low, the extra benefits of any additional WIP will be marginal. Therefore, if there is already sufficient WIP on the shop floor, our policy (unlike those analyzed in [16] or [19]) does not automatically release another unit of WIP to the shop floor every time a finished good is demanded.

In this paper we provide a simple approximate analysis of the base-stock policy for single product multiple stage make-to-stock systems with a limit on the WIP on the shop floor. We also conduct a simulation study that confirms that Veatch and Wein's observations [19] on the effectiveness of base-stock policies extend to systems larger than the two-machine systems they considered. Furthermore, we find that our simple approximation performs very well in estimating the parameters of the best base-stock policy. The rest of this paper is organized as follows. In Section 2 we formulate the optimal control problem as a Markov decision process (MDP), and present details of our proposed base-stock policy. In Section 3 we present a simple approximation method for computing the parameters of the optimal base stock policy. In Section 4 we conduct a simulation study to test how well the proposed base-stock policy works compared with the optimal policy and to test the accuracy of the approximation method developed in Section 3 to estimate the parameters of the optimal base-stock policy. The paper concludes in Section 5.

2. Problem formulation

We consider an N-stage tandem manufacturing system that produces a single product as shown in Fig. 1. There are m_k machines in series in stage k, and the lth machine in stage k has a mean processing time of $t_{k,l}$ (and processing rate $\mu_{k,l} = 1/t_{k,l}$), and standard deviation of $\sigma_{k,l}$. Each unit of inventory held in stage k for a unit of time incurs a holding cost h_k. Raw material is assumed to be available at all times for the first machine in stage 1, and the cost of holding the raw material in front of machine 1 is set to be zero because no value has been added to the material at that point [20]. When a unit's processing is finished at the last machine in stage N, it is transferred to finished goods inventory. Each unit of finished goods inventory incurs a holding cost of h_{N+1} per unit time. Demand is assumed to have a Poisson distribution with

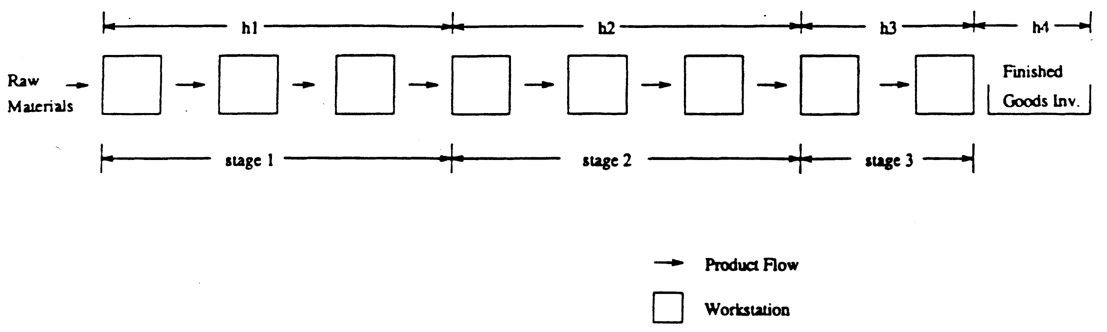

Fig. 1. Tandem production/inventory system.

rate μ_D per unit time. All unfilled demand is backordered and the backordering cost is π per unit time. The objective is to meet the demand with the minimum expected cost per unit time.

In the special case where each stage consists of a single exponential machine, the optimal control problem can be formulated as an MDP [19]. In this case, the state of the system can be represented by an N-vector x. The kth entry ($k = 1, \dots, n$) of x represents the amount of WIP in front of the machine at stage k, whereas x_N is the difference between the amount of finished goods inventory (FGI) in the system and the quantity of backorders. (We will refer to x_N as 'net FGI' and to $x_N^+ = \max\{0, x_N\}$ as 'actual FGI'). We use uniformization and let $\Lambda = \sum_{k=1}^{N} \mu_k + \mu_D$ and e_k denote a unit vector along the kth axis. We can then write the MDP optimality equation as

$$g + V(x) = \frac{1}{\Lambda}\Bigg[c(x) + \mu_D V(x - e_N) + \mu_1 \min\{V(x), V(x + e_1)\} + \sum_{k=2}^{N} \mu_k \min\{V(x), V(x + e_k - e_{k-1})\}\Bigg], \qquad (1)$$

where $c(x) = \sum_{k=1}^{N-1} h_k x_k + h_N x_N^+ - \pi x_N^-$, $x^- = \min\{0, x\}$, $V(x)$ denotes the relative cost of being in state x and following the optimal policy, and g denotes the average cost per transition (where transitions occur with rate Λ) so that $g\Lambda$ gives the optimal average cost per unit time.

Although a similar formulation is possible for the case where each stage has multiple machines in series, the dynamic programming formulation quickly suffers from the curse-of-dimensionality as the number of machines per stage or the number of stages is increased. Furthermore, even for very few stages or machines per stage, the optimal solution has a rather complex structure that makes its implementation very difficult in practice. We therefore focus on simpler base-stock policies.

Our proposed base-stock policy requires only the specification of $N + 1$ nonnegative target inventory (base stock) levels T_1 through T_{N+1}, for implementation. T_k denotes the target sum of inventory in stages k through $N + 1$. Similarly, T_{N+1} denotes the target finished goods inventory level. For example, if $x_{N+1}^+ < T_{N+1}$, this implies that the finished goods inventory level is below target and more finished goods inventory is needed. Therefore, the machines in stage N will keep producing until the finished goods inventory reaches the target level. Similarly, if $\sum_{j=k}^{N} x_j + x_{N+1}^+ < T_k$ this implies that the total inventory downstream from stage k (including stage k) is not sufficient and machines in stage $k - 1$ will keep producing until the level of inventory downstream reaches the target level. Finally, if $\sum_{j=1}^{N} x_j + x_{N+1}^+ < T_1$, this implies that the total amount of inventory in the system is less than the target level and that a new unit of raw material inventory can be released to stage 1. This also implies that the maximum level of inventory in the system is T_1. (We note that this definition of base-stock target levels is slightly different from those in [19] or [14]. Their conditions are of the form $\sum_{j=k}^{N} x_j + x_{N+1} < T_k$. Our conditions on the level of actual inventory in the system provide a way to limit the total amount of inventory on the shop floor.)

We note that the specific case of the above-described base-stock policy, where $T_k = T \; \forall \; k = 1, \dots, N + 1$, corresponds to the make-to-stock version of the CONWIP policy. This policy keeps the total actual inventory in the system constant at all times by releasing a new unit of raw material to the shop floor whenever the total WIP plus actual finished goods inventory in the system falls below T. This job is then pushed through the system. Rubio and Wein [17] propose a version of this policy where the total WIP plus net FGI is kept constant and show that the performance of this policy is easily analyzable under product-form assumptions.

Clearly, to implement the base-stock policy we describe above, the 'optimal' target inventory levels need to be computed. For large systems with many machines per stage, computing the optimal target levels become very difficult. We therefore next present a simple approximation for computing 'approximately optimal' target levels. We then test the performance of the approximation as well as the adequacy of using the proposed base-stock policy.

3. Approximating the base-stock levels

In this section we describe how we approximate the 'optimal' target inventory levels. We first describe the approximation for a one-stage system and then show how it can be generalized to the N-stage case.

3.1. *Single-stage case*

To compute the optimal target inventory levels, we need to be able to compute the cost for any particular choice of target levels. For example, in a single-stage system, in order to set T_1^* and T_2^*, the optimal target levels, we need to be able to compute average WIP, FGI and backorder cost per unit time for any choice of target levels T_1 and T_2. Our approach estimates this cost by approximating each stage by a single equivalent machine.

Consider a single stage system with m_1 machines. Clearly, as long as the last machine has WIP to process, FGI can be produced at the rate of the last machine μ_{1,m_1}. Therefore, as long as the finished goods inventory is below T_2, the last machine in the system will be converting WIP into FGI if it is not starved of WIP. We approximate the rate with which the rest of the machines in the system provide WIP to the last machine by replacing the rest of the machines in the system by a single machine. That is, we replace machines $(1, 1)$ through $(1, m_1 - 1)$ by a single machine. We then have a simpler two-machine system to analyze. The first machine in this simpler system replaces all machines except the last one in the original system, and the second machine is the same as the last machine in the original system. We let $\hat{\mu}_1$ denote the output rate from the first machine to the second machine in this simpler system, and $\hat{\mu}_2$ denote the output rate from the second machine to finished goods inventory. In our approximation, these rates are functions of the inventory levels at the two machines. Let i denote the net finished goods inventory at a given point in time, and j denote the amount of inventory in front of the second machine in this simplified system. Clearly, because the second machine is exactly equivalent to the last machine in the original system, we have

$$\hat{\mu}_2(i,j) = \begin{cases} \mu_{1,m_1} & \text{for } j > 0 \text{ and } i < T_2; \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Equation (2) states the obvious fact that the last machine in the original system will produce parts unless (1) it is starved or (2) the FGI level is equal to the target level. To derive a similar expression for the output rate of the first machine in the simplified system, we first note that in the original system, when the level of net finished goods inventory is negative (i.e., there are backorders) the system behaves like a closed queueing network with T_1 jobs. In this case, whenever the last machine finishes another job, a new job is released to the first machine. What we would like to approximate is the rate with which WIP arrives at the last machine in the original system. Clearly, this depends on the number of jobs at the first $m_1 - 1$ machines. For example, if these machines have no WIP at all, the arrival rate of jobs to the last machine is zero. As the amount of WIP in the first $m_1 - 1$ machines increases, the arrival rate of WIP at the last machine will approach the rate of the slowest machine among the first $m_1 - 1$ machines. We also note that owing to the base-stock policy being used, when the net finished goods inventory (actual finished goods − backorders) is i, and the number of jobs at the last machine is j, the number of jobs in the remainder of the system (i.e., at the first $m_1 - 1$ machines) is $T_1 - i^+ - j$. Combining these observations that the original system behaves like a closed queueing network (at least when $i < 0$) and that the arrival rate of jobs to the last machine is a function of the number of jobs in the first $m_1 - 1$ machines, our simplified system replaces the first $m_1 - 1$ machines with a single machine with rate

$$\hat{\mu}_1(i,j) = TH(T_1 - i^+ - j), \tag{3}$$

where $TH(T_1 - i^+ - j)$ is the throughput of the closed queueing network consisting of the first $m_1 - 1$ machines with $T_1 - i^+ - j$ jobs. We note that when the processing times are assumed to have exponential distributions, this throughput can be computed exactly, using mean value analysis. When the processing times are nonexponential, we use an approximation due to Shanthikumar and Gocmen [21] for approximating the throughput of a closed queueing network with nonexponential machines.

Once we have replaced the original system with a simpler two-station system with rates $\hat{\mu}_1(i,j)$ and $\hat{\mu}_2(i,j)$, we make a further approximation by approximating the processing time distributions at these two stations by an exponential distribution. This simplifies the analysis and, as we show in the next section, the computational results indicate that for highly or moderately variable systems (e.g., processing times with exponential, Erlang-2 or even Erlang-4 distributions), the approximations work very well. For less variable systems, the approach outlined below will tend to overestimate the optimal threshold levels. Hence, in that case, the results can be used as a starting point for a more detailed simulation study.

We let $p(i,j)$ denote the long-run probability of the simplified two-station system having i units of finished goods inventory, and j units of WIP at the second station. Then these probabilities can be easily computed by solving the following system of state equations:

$$\begin{aligned} p(i,j)(\mu_D + \hat{\mu}_2(i,j) + \hat{\mu}_1(i,j)) &= p(i+1,j)(\mu_D) + p(i-1,j+1)\hat{\mu}_2(i-1,j+1) \\ &\quad + p(i,j-1)\hat{\mu}_1(i,j-1). \end{aligned} \tag{4}$$

Once the solution to (4) is obtained, the cost of using target inventory levels T_1 and T_2 can be easily computed as

$$C(T_1, T_2) = \sum_{(i \le T_2; i^+ + j \le T_1)} p(i,j)(h_2 i^+ - \pi i^- + h_1 j). \tag{5}$$

Although (4) represents a system of infinite number of equations (because i can take any integer value below T_2), in practice an approximate solution can easily be obtained by assuming the demand rate falls to zero when the level of finished goods inventory falls below a sufficiently low value. We note that (4) represents a very sparse system of equations, and for a given value of T_1, T_2, a solution can be obtained very quickly. In fact, an exhaustive search for the best threshold values takes just a couple of seconds on a Pentium PC.

3.2. *Multiple stages*

In the case when there is more than one stage, our approach is very similar. In an N-stage problem we replace all the machines in the last stage except the last by a single machine. Similarly, all machines in stage k ($k = 1, \ldots, N-1$) are replaced by a single 'virtual' machine. We thus have a simplified system with $N+1$ machines. Once again, the last machine in the simplified system has a processing rate equal to the last machine in the last stage in the original system. In this case, we can denote the state of the system as an $N+1$-vector, $(i, j_{N+1}, j_N, \ldots, j_2)$, where i denotes the amount of finished goods inventory and j_l denotes the amount of WIP in front of 'virtual machine' l. The processing rate of the last machine is given by

$$\hat{\mu}_{N+1}(i, j_{N+1}, \dots, j_2) = \begin{cases} \mu_{N,m_N} & \text{for } j_{N+1} > 0 \text{ and } i < T_{N+1}; \\ 0 & \text{otherwise;} \end{cases} \tag{6}$$

and for all other machines $k = 1, \dots, N$, it is given by

$$\hat{\mu}_k(i, j_{N+1}, \dots, j_2) = TH_k\left(T_k - i^+ - \sum_{l=k+1}^{N+1} j_l\right), \tag{7}$$

where $TH_k(x)$ is the throughput of a closed queueing network consisting of the machines in stage k and x jobs (note that in computing $TH_N(x)$, we exclude the last machine in stage N). Letting $\underline{j} = (j_{N+1}, \dots, j_2)$, and $p(i, \underline{j})$ denote the long-run probability of the system being in state $(i, \underline{j})$, we need to solve the following system of equations:

$$\begin{aligned} &p(i,\underline{j})\left(\mu_D + \sum_{k=1}^{N+1} \mu_k(i,\underline{j})\right) \\ &\quad = p(i+1,\underline{j})\mu_D + \mu_{N+1}(i-1,\underline{j}+e_{N+1})p(i-1,\underline{j}+e_{N+1}) \\ &\qquad + \left(\sum_{k=2}^{N} p(i,\underline{j}-e_{k+1}+e_k)\mu_k(i,\underline{j}-e_{k+1}+e_k)\right) \\ &\qquad + p(i,\underline{j}-e_2)\mu_1(i,\underline{j}-e_2). \end{aligned} \tag{8}$$

We let

$$\varphi = \{(i,\underline{j}) : i \le T_{N+1}; i^+ + \sum_{l=k+1}^{N+1} j_l \le T_k \quad \text{for } k = 1, \dots, N\}; \tag{9}$$

the cost of using target inventory levels $(T_1, \dots, T_{N+1})$ is then given by

$$C(T_1, \dots, T_{N+1}) = \sum_{\varphi} p(i,\underline{j})\left(h_{N+1}i^+ - \pi i^- + \sum_{k=2}^{N+1} h_{k-1}j_k\right). \tag{10}$$

Once again, the system of equations (8) can be solved rapidly owing to the sparsity of the system of equations. We next report on the quality of the solutions obtained by this approximation technique in terms of estimating the best threshold values as well as on the performance of the proposed base-stock policy compared with the optimal policy.

4. Computational results

This section reports the results of a simulation study that we conducted to test the performance of the proposed base-stock policy as well as the success of the approximation scheme outlined in the previous section. To conduct our study, we generated 50 example cases representing a wide variety of situations. We created examples with three different processing time distributions: exponential, Erlang-2, and Erlang-4, to capture the effects of different levels of variability on the approximation. We also created examples with varying levels of line length and numbers of stages as well as different holding and backorder costs. We tested examples where the backorder cost was higher than the finished goods inventory cost as well as the case when the reverse was true. For each example, we used simulation to find the best base-stock policy by using a GPSS-H program. For each candidate vector of base-stock values, we obtained the simulation value of the cost by running 20 replications of simulations 10 days in length each (after deleting the warm-up period). The approximation described in Section 3 was also used to compute the 'approximately optimal' base-stock values. That is, we used the approximation to compute the cost for each candidate base-stock policy and chose the best base-stock policy found by the approximation. We then used simulation to compute the cost of the base-stock policy suggested by the approximation. The cost difference between the cost of the best base-stock policy found by simulating all feasible base-stock policies and the cost of the base-stock policy suggested by our approximation is the measure of the approximation's success. Finally, we also report the average cost achieved by the best CONWIP policy. We note that because the CONWIP policy is a special case of the base-stock policy, the best base-stock policy is guaranteed to perform at least as well as the best CONWIP policy.

Table 1 includes the data and the optimal solutions obtained by solving the MDP for Examples 1–27. We note that all of the processing times as well as the time between two consecutive demands are assumed to have exponential distributions in Examples 1–27. The cost of holding inventory in front of each machine (starting with the second machine as the cost of holding raw material in front of machine 1 is set to 0) is given by the vector h. The last entry of h is the cost of finished goods inventory. For example, in Example 1, the cost of holding WIP is assumed to be 1, and the FGI cost is 3 per item per unit time. In Examples 1–21 the cost of WIP is the same regardless of the location of WIP. In Examples 22–27 the cost of WIP changes from machine to machine. However, the value added is very small except at the very last operation. Therefore, in computing the base-stock values, we tested the performance of an approximation policy that treats the cost of WIP as the same at each of these operations. For instance, the WIP costs in Example 27 are 0.7, 1, and 1.3 for WIP waiting at machines 2, 3, and 4. Therefore, to be able to reduce this problem to a single-stage problem in our approximation, we assumed WIP costs to be 1. These examples test the performance of our strategy, which replaces several stages with little value added at each stage by a single stage with the average WIP cost of the original stages.

Table 1. Input data and the optimal cost for Examples 1–27

Example	*Processing times*	μ_D	*h*	π	*Optimal*
1	2,3,4,3	5	1,1,1,3	5	19.09
2	3,2,3,4	5	1,1,1,3	5	21.10
3	3,3,4,2	5	1,1,1,3	5	16.66
4	3,4,3,4	5	1,1,1,3	5	27.70
5	2,2,3,4	5	1,1,1,3	5	20.29
6	1,1,3,4	6	1,1,1,1.5	2	6.61
7	3,3,4	5	1,1,3	5	20.71
8	2,3,4	5	1,1,3	5	19.62
9	2,3,4	5	1,1,3	2	12.74
10	3,4,2	5	1,1,3	5	14.29
11	3,4,2	5	1,1,3	2	10.14
12	2,4,3	5	1,1,3	5	16.43
13	2,4,3	5	1,1,3	2	11.20
14	1,1,3	4	1,1,1.5	2	6.84
15	2,2,4	5	2,2,3	4	19.03
16	1,2,3	4	1,1,2	3	10.49
17	3,3,2	5	4,4,5	7	22.58
18	2,4,4	7	2,2,2.5	3	10.43
19	3,5,5	7	2,2,3	5	22.18
20	2,5,5	7	1,1,1.5	3	11.31
21	3,4,1	5	2,2,4	5	18.09
22	2,3,4,3	5	0.5,1,1.5,3	5	20.73
23	3,2,3,4	5	0.5,1,1.5,3	5	22.17
24	3,3,4,2	5	0.5,1,1.5,3	5	18.12
25	3,4,3,4	5	0.5,1,1.5,3	5	28.62
26	2,2,3,4	5	0.5,1,1.5,3	5	21.39
27	1,1,3,4	6	0.7,1,1.3,1.5	2	6.87

Table 2. Performance of base-stock and CONWIP policies for Examples 1–27

Example	*Best base-stock*	*App. base-stock*	*CONWIP*	*ABS/BBS (%)*
1	(11,3) 4.6%	(11,2) 7.3%	(10) 29.6%	2.6
2	(10,5) 7.3%	(9,5) 9.0%	(10) 10.2%	1.6
3	(12,0) 4.9%	(12,1) 6.7%	(10) 42.4%	1.7
4	(16,5) 3.0%	(14,5) 7.3%	(13) 13.8%	4.1
5	(10,5) 4.5%	(8,5) 5.9%	(8) 12.8%	1.3
6	(4,3) 7.7%	(4,3) 7.7%	(4) 7.8%	0.0
7	(10,5) 3.6%	(9,5) 7.9%	(7) 9.7%	4.1
8	(9,6) 3.0%	(7,5) 6.6%	(7) 8.4%	2.0
9	(7,3) 7.9%	(6,3) 11.8%	(6) 11.7%	3.6
10	(9,1) 7.4%	(10,1) 11.1%	(7) 40.1%	3.4
11	(8,0) 10.9%	(7,0) 15.8%	(6) 36.0%	4.4
12	(9,2) 7.2%	(9,2) 7.2%	(7) 28.6%	0.0
13	(7,1) 11.6%	(7,1) 11.6%	(6) 27.2%	0.0
14	(4,4) 3.2%	(3,3) 7.4%	(4) 3.2%	4.0
15	(6,6) 0.9%	(5,5) 1.7%	(6) 0.9%	0.8
16	(5,5) 1.3%	(5,4) 3.3%	(5) 1.3%	2.0
17	(5,4) 12.0%	(5,2) 12.8%	(5) 15.4%	0.7
18	(4,3) 6.7%	(4,3) 6.7%	(4) 8.1%	0.0
19	(7,6) 2.9%	(7,7) 3.9%	(7) 3.9%	0.9
20	(7,7) 4.7%	(7,7) 4.7%	(7) 4.7%	0.0
21	(7,0) 10.8%	(7,0) 10.8%	(6) 28.5%	0.0
22	(10,4) 0.9%	(11,2) 1.5%	(10) 13.4%	0.6
23	(9,8) 1.8%	(9,5) 8.1%	(9) 11.3%	6.2
24	(11,1) 0.8%	(12,1) 5.5%	(10) 24.8%	4.7
25	(13,4) 0.1%	(14,5) 7.7%	(13) 12.3%	7.6
26	(9,8) 0.3%	(8,5) 3.6%	(10) 12.1%	3.3
27	(4,4) 6.1%	(4,3) 8.6%	(4) 6.1%	2.4

The results for Examples 1–27 are displayed in Table 2. For each example, Table 2 includes the best base-stock values found by simulation and the percentage suboptimality of the best base-stock policy compared with the optimal policy found by solving the MDP. Similarly, Table 2 includes the base-stock values computed by using the approximation we presented in the previous section and the suboptimality of this policy. Table 2 also displays the percentage cost difference between the cost of the best base-stock policy found by simulation, and that found by our approximation (ABS/BBS). Finally, the best CONWIP policy as well as its suboptimality are also given for each example. For example, the first entry in Table 2 indicates that the best base stock policy found by simulation had the base-stock values (11,3). The suboptimality of the best base-stock policy was 4.6%. This value was found by comparing the value of the optimal policy found by solving the MDP with the value obtained by simulating the base-stock policy with the base-stock values (11,3). When the approximation described in the previous section was used to find the best base-stock policy, the best policy found by the approximation was (11,2), which had a suboptimality of 7.3%. (These values are reported under the 'App. base-stock' column). The suboptimality value was also found by comparing the optimal solution value obtained by the MDP with the cost obtained from the simulation of the base-stock policy (11,2). The ABS/BBS column shows that the best base-stock policy suggested by the approximation, (11,2), resulted in a 2.6% higher cost than the best base-stock value obtained by simulation, (11,3). Finally, the best CONWIP policy and its suboptimality, found by simulation, is reported under the CONWIP column. We note that the CONWIP policy is a special case of the base-stock policy; therefore the best base-stock policy will perform no worse than the best CONWIP policy. Examples 14, 15, 16, and 20 are some cases where the best base-stock policy is of the CONWIP form and therefore the suboptimality reported under both columns is the same for these cases.

The results in Table 2 show that the best base-stock policy performs very well compared with the optimal policy. The average suboptimality of the best base-stock policy found by simulation is around 5%. We note that the optimal policy is very complex whereas the base-stock policy is very simple to describe and implement. Given that it is hard to estimate backorder or holding costs to a precision of 5%, the performance of the simple base-stock policy is encouraging. Table 2 also shows that our approximation behaves very well. The average suboptimality of the base-stock policy with the target inventory levels computed by using our approximation was 7.5%. As Table 2 clearly demonstrates, our approximation's estimates of the best base-stock levels were very close to the optimal base-stock levels found by simulation. In fact, the average cost difference between the best base-stock policy and that suggested by our approximation was about 2.5%. Given the speed with which our approximation computed the best base-stock values, these results are encouraging.

Table 3 includes the data for Examples 28–50. In Examples 28–38, the processing times are still exponential. However, these examples have multiple stages and as many as seven machines. We were unable to compute the optimal costs owing to the very large size of the state spaces in these examples. In Examples 39–46 the processing times had Erlang-2 distributions, whereas in Examples 47–50 the distributions were Erlang-4. As described in the previous section, our approach in these cases was the same as in the exponential cases except that we used an approximation due to Shanthikumar and Gocmen [21] for computing the throughputs of the nonexponential closed queueing networks involved.

The results for Examples 28–50 are presented in Table 4. Table 4 includes the best base-stock values found by simulation and the cost associated with the best base-stock policy, the base-stock values suggested by our approximation and the percentage cost difference between the best base-stock policy and the base-stock policy computed by the approximation. We also present the best CONWIP policy and its percentage difference from the best base-stock policy. For example, the first entry in Table 4. indicates that for Example 28 the best base-stock policy found by simulation was (6,5,1), which had an average cost per unit time of 9.19. When we used the approximation to compute costs for all base-stock policies, the best base-stock policy found by the approximation was (6,6,1) and this policy resulted in a 3.0% higher cost than the best base-stock policy. Finally, the best CONWIP policy resulted in a 27.4% higher cost than the best base-stock policy. (We remind the reader that whereas in Table 2 the percentage values reported represent the suboptimality of each policy with respect to the optimal policy, the percentage values reported in Table 4. represent the percentage difference between the cost of a policy and the cost of the best base-stock policy. This is because we were unable to solve the MDP to obtain optimal costs for Examples 28–50).

We note that despite the fact that these examples were more challenging to our approximation (owing to either the greater size of the problems or the non-exponential distributions involved), our approximation still performed very well in estimating the best base-stock parameters. In the examples with Erlang distributions, our approximation predictably tended to overestimate the target inventory levels. These examples indicate that if the processing times distributions are less variable than exponential, the results of our approximation can serve as approximate

Table 3. Input data for Examples 28–50

Example	*Pr. times*	*Distribution*	μ_D	*h*	π
28	1,2,3,1,2	Exp.	5	0.5,0,5,1,1,3	5
29	2,3,1,2,3	Exp.	5	0.5,0.5,1,1,3	5
30	1,3,2,1,3	Exp.	5	0.5,0.5,1,1,3	5
31	1,2,3,4,1,2	Exp.	5	0.5,0.5,0.5,1,1,3	5
32	2,2,4,3,3,1	Exp.	5	1,1,1,2,2,3	6
33	3,1,2,2,1,1	Exp.	6	1,1,1,3,3,5	7
34	1,2,3,4,5,1,2	Exp.	7	0.5,0.5,0.5,1,1,1,3	5
35	2,1,3,1,4,1,4	Exp.	6	1,1,1,2,2,2,3	6
36	5,1,2,1,1,3,3	Exp.	7	2,2,2,3,3,3,4	4
37	4,3,4,3,1,4,2	Exp.	7	2,2,2,3,3,3,3.5	4
38	4,4,3,1,2,1,2	Exp.	6	1,1,1,5,5,5,6	6
39	2,3,4,3	Erl-2	5	1,1,1,3	5
40	3,3,4,2	Erl-2	5	1,1,1,3	5
41	2,2,3,4	Erl-2	5	1,1,1,3	5
42	2,3,4	Erl-2	5	1,1,3	5
43	3,4,2	Erl-2	5	1,1,3	5
44	3,4,2	Erl-2	5	1,1,3	2
45	2,4,3	Erl-2	5	1,1,3	5
46	2,4,3	Erl-2	5	1,1,3	2
47	2,3,4,3	Erl-4	5	1,1,3	5
48	3,3,4,2	Erl-4	5	1,1,3	5
49	1,4,4,3	Erl-4	5	1,1,2	3
50	2,1,2,1	Erl-4	3	2,2,3	4

Table 4. Results for Examples 28–50

Example	*Best base-stock*	*App. base-stock*	*CONWIP*
28	(6,5,1) 9.19	(6,6,1) 3.0%	(5) 27.4%
29	(8,7,2) 12.38	(7,7,2) 2.9%	(7) 16.3%
30	(7,6,2) 12.22	(6,6,2) 0.7%	(6) 12.9%
31	(13,11,0) 15.9	(10,10,1) 11.3%	(8) 40.4%
32	(11,11,0) 26.3	(10,10,1) 11.2%	(12) 11.9%
33	(5,3,0) 13.0	(4,4,0) 3.0%	(4) 19.8%
34	(10,10,0) 12.8	(9,9,1) 7.5%	(8) 36.8%
35	(8,8,8) 22.9	(8,8,4) 3.6%	(8) 0.0%
36	(6,5,2) 18.9	(5,5,2) 12.1%	(6) 7.10%
37	(8,7,3) 24.1	(7,7,2) 6.3%	(8) 4.2%
38	(10,4,1) 26.3	(8,5,2) 10.6%	(8) 21.8%
39	(8,2) 14.9	(11,2) 7.6%	(8) 21.4%
40	(10,1) 13.5	(12,1) 9.2%	(7) 30.3%
41	(7,4) 15.8	(7,7) 6.6%	(6) 4.4%
42	(6,3) 15.1	(5,5) 2.6%	(5) 2.6%
43	(7,1) 12.0	(9,1) 3.2%	(6) 14.2%
44	(5,0) 8.7	(7,0) 5.7%	(5) 22.9%
45	(7,2) 13.7	(9,2) 2.9%	(6) 13.9%
46	(5,1) 9.1	(6,1) 4.4%	(4) 15.4%
47	(6,2) 12.2	(9,2) 8.2%	(5) 16.3%
48	(7,1) 11.1	(8,1) 2.7%	(6) 27.3%
49	(7,2) 11.1	(8,3) 10.1%	(7)12.6%
50	(4,1) 11.4	(4,1) 0%	(4) 4.4%

upper bounds on the amount of inventory required at each stage, and therefore as a starting point for a more detailed simulation study. However, we note that even using the values suggested by the approximation resulted in costs that were not much higher than the cost of the best base-stock policies. The average percentage cost difference between the best base-stock policy and that suggested by our approximation was around 6%. In contrast, the average percentage difference in cost between the best CONWIP policy and the best base-stock policy was nearly 17%. These results clearly show the significant decrease in cost that can be obtained by using multiple-stage base-stock policies and the success of our approximation, which involves very little computational work.

5. Conclusions and further research

In this paper we analyzed base-stock policies for multiple-stage tandem make-to-stock systems. The base-stock policies we analyzed differ from those in the literature in that they limit the WIP on the shop floor, therefore avoiding unnecessary stockpiling of WIP on the shop floor when there are many backorders. Our simulation results comparing the performance of the proposed base-stock policies with the performance of the optimal policy indicate that the proposed base-stock policies perform very well. Comparisons of the cost of the optimal policy and the cost of the best base-stock policy in our simulation experiments reveal that Veatch and Wein's observations [19] on the effectiveness of base-stock policies (which were based on a limited set of experiments with only two machines in their paper) carry over to larger and more complicated systems. We also presented a simple approximation based on aggregation of several stages (with the same or very close WIP costs) into one for computation of 'approximately optimal' base-stock levels. We presented the results of a simulation study that demonstrated that our approximation is successful in estimating the best base-stock values.

Further research should characterize effective and simple control strategies for more complicated systems than those considered here. Examples include multiple-stage assembly systems where the output of several sub-assembly lines are assembled together, and systems with probabilistic routing of products.

Acknowledgements

We are grateful to an anonymous referee for his/her suggestions, which greatly improved the presentation and clarity of the paper. This research is partly supported by NSF Grants DMI-9308290 and DMI-9424596 to the University of Michigan, and by a grant from the Center for Display Manufacturing.

References

[1] Mitra, D. and Mitrani, I. (1990) Analysis of a kanban discipline for cell coordination in production lines. I. *Management Science*, **36**, 1548–1566.

[2] Muckstadt, J.A. and Tayur, S.R. (1995) A comparison of alternative kanban control mechanisms, I, II. *IIE Transactions*, **27** (2), 140–161.

[3] Mitra, D. and Mitrani, I. (1991) Analysis of a kanban discipline for cell coordination in production lines. II. Stochastic demands. *Operations Research*, **39**, 807–823.

[4] Duenyas, I. and Keblis, M. (1995) Release policies for assembly systems. *IIE Transactions*, **27**, 507–518.

[5] Chang, T.M. and Yih, Y. (1994) Generic kanban systems for dynamic environments. *International Journal of Production Research*, **32** (4), 889–902.

[6] Chang, T.M. and Yih, Y. (1994) Determining the number of kanbans and lotsizes in a generic kanban system: a simulated annealing approach. *International Journal of Production Research*, **32**, (8), 1991–2004.

[7] Askin, R.G., Mitsawi, M.G. and Goldberg, J.B. (1993) Determining the number of kanbans in multi-item just-in-time systems. *IIE Transactions*, **25**, (1), 89–98.

[8] Mitsawi, M.G. and Askin, R.G. (1994) Production planning for a multi-item, single-stage kanban system. *International Journal of Production Research*, **32** (5), 1173–1195.

[9] Spearman, M.L., Hopp, W.J. and Woodruff, D.L. (1990) CONWIP: a pull alternative to kanban. *International Journal of Production Research*, **28**, 879–894.

[10] Hopp, W.J. and Spearman, M.L. (1991) Throughput of a constant work in process manufacturing line subject to failures. *International Journal of Production Research*, **29**, 635–655.

[11] Duenyas, I., Hopp, W.J. and Spearman, M.L. (1993) Characterizing the output process of a CONWIP line with deterministic processing and random. *Management Science*, **39**, 975–988.

[12] Duenyas, I. and Hopp, W.J. (1993) Characterizing the output process of a CONWIP line with deterministic processing and random outages. *Management Science*, **39**, 975–988.

[13] Duenyas, I. (1994) Estimating the throughput of a cyclic assembly system. *International Journal of Production Research*, **32**, 1403–1419.

[14] Buzacott, J.A., Price, S. and Shanthikumar, J.G. (1992) Service levels in multistage MRP and base stock controlled production systems, in *New Directions for Operations Research in Manufacturing*, Fandel, G., Gulledge, T. and Jones, A. (eds), Springer, Berlin, pp. 445–463.

[15] Buzacott, J.A. and Shanthikumar, J.G. (1993) *Stochastic Models of Manufacturing Systems*, Prentice-Hall, Englewood Cliffs, NJ.

[16] Lee, Y.J. and Zipkin, P. (1992) Tandem queues with planned inventories. *Operations Research*, **40**, 936–947.

[17] Rubio, R. and Wein, L.M. (1996) Setting base stock levels using product-form queueing networks. *Management Science*, **42** (2), 259–268.

[18] Uzsoy, R., Lee, C.Y. and Martin-Vega, L.A. (1994) A review of production planning and scheduling models in the semiconductor industry, Part II. Shop floor control. *IIE Transactions on Scheduling and Logistics*, **26**, 44–55.

[19] Veatch, M.H. and Wein, L.M. (1994) Optimal control of a two-station tandem production/inventory system. *Operations Research*, **42**, 337–350.

[20] Duenyas, I. (1994) A simple release policy for networks of queues with controllable inputs. *Operations Research*, **42**, 1162–1171.

[21] Shanthikumar, J. and Gocmen, M. (1983) Heuristic analysis of closed queueing networks. *International Journal of Production*, **21**, 675–690.

Biographies

Izak Duenyas is Associate professor of industrial and operations management at the University of Michigan. He received his Ph.D. in industrial engineering from Northwestern University in 1991. Dr. Duenyas's research interests are in the analysis and control of manufacturing systems. He serves on the editorial boards of *Operations Research, MSOM Journal, IIE Transactions on Scheduling and Logistics,* and *International Journal of Flexible Manufacturing Systems*.

Prayoon Patana-Anake is a doctoral student in the IOE department at the University of Michigan.