Setting Basestock Levels in Multiproduct Systems with Setups and Random Yield

Scott E. Grasman [1]

grasmans@umr.edu

219 Engineering Management

University of Missouri - Rolla, Rolla, Missouri 65409-0370

Tava Lennon Olsen

olsen@olin.wustl.edu

John M. Olin School of Business, Campus Box 1133

Washington University in St. Louis, St. Louis, Missouri 63130-4899

John R. Birge

jrbirge@nwu.edu

McCormick School of Engineering and Applied Science

Office of the Dean, 2145 Sheridan Road

Northwestern University, Evanston, Illinois 60208

March 2003

[1]Corresponding Author

Abstract

This paper shows how to set basestock levels in a multiproduct system with setups and random yields. The system is represented by a polling model and the inventory level of each product is controlled using a basestock policy. When the queue is empty, the inventory level is equal to the basestock level, thus the server will continue to serve the queue until it is empty. If the capacity of the queue is equal to the inventory allocated to the item, then when the queue is full, inventory is fully depleted and new demand is either backlogged, lost or expedited. Defects are routed to temporary storage queues associated with each item, and then routed back to the original queue for service during the next cycle. For a system with backlogging, we provide a cost function and show that it is minimized by solving N single item newsvendor problems. For a system with lost sales or expediting, we introduce a cost function and provide a heuristic for finding the basestock levels. The heuristic is then demonstrated via an example.

Keywords: Basestock, random yield, multiproduct, setups

1 Introduction

Most production systems are characterized by multiple uncertainties, which cause output to be highly variable. Two major contributors to the uncertainty are variable capacity and random yield. When production capacity is variable, the realized capacity in a given time period could constrain the quantity that can actually be processed. Therefore, if the planned production is greater than the realized capacity, only part of the lot will be produced. When yield is random, not only is the usable quantity uncertain, but the capacity and throughput of the system are also reduced. For these reasons, production strategies are needed which incorporate variable capacity and random yield. This paper addresses a multiproduct system with significant setups and random yield. The particular multiproduct system we consider is motivated by a producer of plastic bumpers for automobiles that we visited often, which we refer to as *BumperCo*.

BumperCo has a single paintline for its bumpers producing yields of around 75 percent. Defects that occur during the painting process must be reworked and repainted. Furthermore, changeovers between different paint colors are expensive and time consuming; hence, products are produced in batches of the same color. The painting process is approximately six hours from start to finish, which eliminates the possibility of inspecting units before starting the next lot. Specific characteristics of this process are that(a) yield is highly unpredictable; therefore, it is not known ahead of time how many good parts will be produced in any given production run; (b) successive defects are independent (as the majority of defects are caused by dirt, dust, and scratches, which occur approximately independently); and (c) it is not known how many defects are produced in a given batch until after the next batch has been started. This means that it is prohibitively expensive to immediately remake the product just produced if its yield is lower than expected; therefore, this problem differs from the situation where the decision whether to continue the current lot can be made after each unit of production. Demand for each item is random, and, if not met, is either backlogged, lost, or expedited at varying costs. Possible expediting includes outsourcing, overtime production, and express delivery. Also associated with each item are random production and setup times, random yields, inventory holding costs, and penalty costs due to shortage. For

the case of lost sales or expediting, the analysis does provide service levels (average fraction of lost/expedited customers) and a service level approach could be used, the service level approach would not take into account the significant shortage costs present in this context. Additionally, we consider processing time, and not just costs, since random yield can have a great impact on the capacity of the production system.

We do not attempt to model the "seat of the pants" scheduling system we found in place at BumperCo. Instead, we seek to find optimal (minimum cost) basestock levels under the type of random yield found at BumperCo. While such a policy is not generally optimal, it is easy to implement and often performs very well (see Federgruen and Katalan [5] for further motivation on such production systems). Under such a policy, when inventory position is reviewed and found to be below S, production continues until the inventory position (including defects in this case) reaches the basestock level, S. After completion of service, the server moves in a periodic pattern, meaning that the server serves queues 1 to N and then returns to queue 1, which is typical of many fixed cycle production schedules (see, Elmaghraby [4]). Our objective is similar to that of Federgruen and Katalan [5] with random yields added. They show that the optimal production strategy under a cyclic basestock policy with backlogging corresponds to a newsvendor solution. We will be able to follow a similar technique for our system with backlogging, however, this policy does not work for lost sales or expediting. Additionally, calculation of the optimal policy for lost sales or expediting is only tractable for small problems, thus we will introduce a heuristic to search for optimal basestock levels.

Random yields are, of course, found in broader contexts than that of BumperCo. Indeed, many modern production processes are substantially affected by random yields. Semiconductor manufacturers, for instant, often have yields below 50 percent (see, e.g., Bohn and Terwiesch [2] and Gilbert and Emmons [6]). Further, production processes which are new frequently have yields that are well below the ideal. As more is learned, yields increase, but generally never reach 100 percent (or even come close in some cases). Rapidly changing technologies may become obsolete before the process is well understood, or it may not be financially justifiable to correct the yield problem. Therefore, in these situations, methods are needed to cope with uncertainty. Yano and Lee [17] point out that while the ultimate

goal is to make yields perfect, random yield models are valuable for many reasons. First, results can be used in the short term to help an operation run more efficiently so that efforts can be focused on yield improvement. Second, process improvement and supplier decisions can be assessed more accurately and effectively if system effects of these decisions are modeled accurately and optimized. Third, models can assist in capacity planning decisions when random yield is expected to be a long run concern.

The subject of process yields has received considerable attention, which is not surprising considering its significance across manufacturing industries. Numerous papers have been written regarding production/inventory models with random yield; however, few deal with multiple product capacitated systems. Yano and Lee [17] provides a comprehensive review of the literature on quantitatively-oriented approaches for determining lot sizes when production or procurement yields are random. Issues discussed relate to the modeling of costs, yield uncertainty, and performance. Existing classical production approaches may be modified to consider average yield rates by simply adjusting the production rates and normalizing the production costs per unit to account for average yield losses, but this approach is insufficient. Such approaches do not capture the fact that, in the presence of random yield, it is usually desirable to produce a greater quantity due to the fact that shortage costs are greater than holding costs. These approximations yield optimal results only if the yield is deterministic. Bollapragada and Morton [3] discuss these approaches at length.

Krieg and Kuhn [12] proposes a method for determining the setup cycle and number of kanbans for a multiproduct system with setup times and lost sales. For the same system, Krieg and Kuhn [11] presents a decomposition-based approximation method for estimating steady-state performance measures, and Krieg and Kuhn [13] extends the results to state-dependent setup times. Although similar to this paper, they differ in two main areas: 1) they do not allow for routing of defective products, and 2) the objective only considers minimizing holding costs based on the average inventory level while achieving a required fill rate, and does not consider shortage costs (or rework costs). Rajaram and Karmarkar [15] presents a heuristic policy for discrete time version of the dynamic product-cycling problem with random yield and demonstrate its application to process industries.

This paper uses the previous polling model analysis of Grasman [7] and Grasman et

al. [8] to set basestock levels in multiproduct systems with setups and random yields and is organized as follows. Section 2 details the basic model and assumptions. The infinite buffer system, which corresponds to the system with backlogging, is presented in Section 3, including the development and optimization of the cost function, the derivation of the optimal basestock policy, and an illustrative example. The finite buffer system, corresponding to the system with lost sales or expediting is presented in Section 4, including the development of the cost function, a heuristic basestock policy, and numerical examples. A summary and future work are provided in Section 5.

2 Problem Formulation

This paper sets basestock levels in multiproduct systems with setups and random yields where all products must be produced on a single machine. We consider a continuous time version of this problem by representing the production facility as a standard queueing model and the inventory of each product by a classical model from inventory theory. For each product, batches are produced according to a basestock policy, specifically, an $(S-1, S)$ policy; when inventory position is reviewed and found to be below S, production continues until the inventory position, which is inventory on hand plus items produced (including defects) minus items backlogged, reaches the basestock level, S. After completion of service, the server moves in a periodic pattern, meaning that the server serves queues 1 to N and then returns to queue 1, which is typical of many fixed cycle production schedules (see, Elmaghraby [4]). Although Henig and Gerchak [9] has shown that the optimal policy for the random yield problem is not order-up-to, basestock policies are easily implementable and are often used in practice. The production facility is represented as one server in a queueing system and only one type of product can be produced at a time, thus this type of model is a polling model.

The system considered in this paper is depicted in Figure 1 and consists of a single server and N queues representing $\frac{N}{2}$ items. Since each "real" queue is followed by a temporary storage queue, the queue corresponding to item i is indexed $2i-1$ ($2i-1$ is odd), while the temporary storage queue for item i is indexed $2i$ ($2i$ is even). Since the yield of each item

is assumed to be independent of all others, each item either exits the system or is routed to a temporary storage queue according to the Bernoulli distribution. Note that defects are stored in temporary storage queues and not immediately routed back to the original queue in order to account for the delay between production and inspection; thus, a policy of producing at least enough items to account for the defects is not possible. Once the server completes service of a particular item, defects are routed back to the original queue for service during the next cycle.

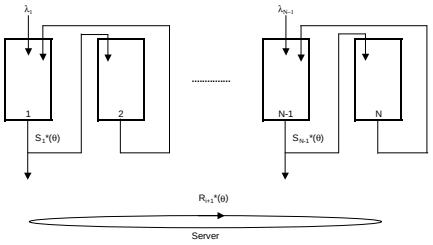

Figure 1: Polling Model Representation

There is no setup required for the storage queues and also no service time; therefore, the storage queues do not contribute to the utilization of the system. Items are routed from queue i to queue j ($j = i+1$) with probability p_{ij} due to defective production. For the finite buffer system, routed items that find the queue full are lost. After the inventory position of an item reaches its basestock level, the server then moves to the storage queue and routes the defective items back to their original queue. Thus, $p_{ij} = 1 - y_i$ and $p_{ji} = 1$ for $j = i + 1$ and i odd, where y_i is the yield rate. For $j \neq i + 1, p_{ij} = 0$.

The external arrivals to each "real" queue, $2i-1, 1 \leq i \leq \frac{N}{2}$, are assumed to be according to a Poisson process with rate λ_i, corresponding to the demand of each product. The service at any queue follows a general distribution with Laplace-Stieltjes Transform (LST) $S_i^*(\theta)$, $1 \leq i \leq N$, and exhaustive service discipline, meaning that the server continues to serve until the queue is empty. Thus, all demand present at the beginning of service and all that arrives during the service period is produced in that period. When the queue is empty, the inventory level has been replenished to its basestock level. After completion of service at both the real

and storage queues, the server moves in a cyclic fashion and incurs a switchover period whose duration corresponds to the setup of the next item, and follows a general distribution with LST $R_i^*(\theta)$, $1 \leq i \leq N$. The process of the server arriving at a queue for service is know as polling, and the time of polling is referred to as a polling instant. Although, state-independent switchovers are assumed, the analysis is intended for systems with significant setup times, high utilization, and significant random yield. In order for a queue to be empty upon polling, there would have to be no demand and no defective production during the previous cycle, thus it is unlikely that state-independent setups would have a major impact under the intended application. The inventory level of an item i, $1 \leq i \leq \frac{N}{2}$, is equal to the basestock level, L_i, minus the queue length of both the real and storage queues, denoted Q_{2i-1} and Q_{2i}, respectively. Therefore, if backlogging is allowed, the queues are given infinite buffers, and the basestock level subtracted from the queue length (if positive) is the number of backorders on hand. If backorders are not allowed, then the queue is given a finite buffer, $\bar{K}$, equal to the basestock level; therefore, when the queue is full, new demand is lost or expedited.

Since the occurrences of defects during a production run are assumed to be independent and identically distributed, the number of good units in a batch of size Q is binomially distributed with parameters Q and y, where y is the probability of producing a good unit. One advantage of this model is that only the value of y_i needs to be specified. One disadvantage is that it does not allow for specification of variance. The expected value of the fraction of good units produced remains constant (y), but the variance of the fraction of good units, $\frac{y(1-y)}{Q}$, decreases with Q, which is a significant characteristic of the binomial model. This modeling approach is appropriate for the painting process because the system is in control for long periods of time and each unit produced may be considered independent of the previous unit.

Previous Results

Grasman [7] extends the results of Sidi et al. [16] and presents the analysis of an infinite buffer polling system with routing to a temporary storage queue, which corresponds to the random yield production model with backlogging. Grasman et al. [8] extends the results of

Jung and Un [10] and presents the analysis of a finite buffer polling system with routing. This system may be used to analyze a random yield production system with lost sales or expediting. The most significant result of the analysis is the derivation of the generating function for queue length when buffer sizes are limited. The buffer occupancy variables, which represent the number of customers present at each queue at a polling instant, are determined by solving a system of recursive equations. An embedded Markov chain analysis and numerical inversion were used to derive the queue length distribution.

3 Systems with Backlogging

This section addresses optimal basestock levels for infinite buffer systems, which correspond to the system with backlogging shown in Figure 2.

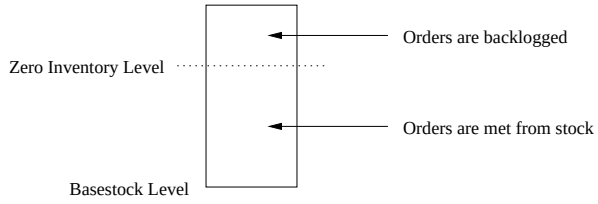

Figure 2: Queueing Representation of Backlogging

We first introduce the cost functions that are used and provide a theorem which allows us to determine the optimal basestock levels using the newsvendor model. Finally, we derive an expression for the optimal basestock levels for arbitrary convex and nondecreasing cost functions and provide an example.

3.1 Cost Function

Costs include setup costs, holding costs, and shortage costs. Since defects are produced at a constant rate, the cost of rework is constant and omitted from the cost function. However, since defective items remain in the system, they still contribute to holding and shortage costs, as well as production times. We assume that the system with $\frac{N}{2}$ items is in steady-state, and let

$$I_i \quad = \quad \text{inventory level of item } i, \text{ and } I = (I_1, I_2, ..., I_{\frac{N}{2}}),$$

L_i = basestock level of item i, and $L = (L_1, L_2, ..., L_{\frac{N}{2}})$,

Q_i = queue length of queue i, and $Q = (Q_1, Q_2, ..., Q_N)$, and

c = cycle time, which is the time required to compete service at all queues.

Since both the "real" and temporary storage queues contribute to the inventory level of an item, the inventory level of item i, is $I_i = L_i - Q_{2i-1} - Q_{2i}$ for $1 \leq i \leq \frac{N}{2}$. Since backlogging is allowed and the buffer is assumed to be infinite, the number of Poisson arrivals to any queue is independent of the basestock level at any queue. Thus Q_i is independent of L_j for all i, j.

Also define

c_i^r = setup cost of item i,

$h_i()$ = convex and nondecreasing holding cost function for item i, and

$\pi_i()$ = convex and nondecreasing shortage cost function for item i.

Then,

$$\begin{aligned} C(L) &= \text{long run average cost under basestock policy, } L \\ &= \frac{\sum_{i=1}^{\frac{N}{2}} c_i^r}{c} + \sum_{i=1}^{\frac{N}{2}} E[h_i(I_i^+) + \pi_i(I_i^-)] \\ &= \frac{\sum_{i=1}^{\frac{N}{2}} c_i^r}{c} + \sum_{i=1}^{\frac{N}{2}} E[h_i((L_i - Q_{2i-1} - Q_{2i})^+) + \pi_i((L_i - Q_{2i-1} - Q_{2i})^-)], \end{aligned}$$

where $I^+ = max\{I, 0\}$ and $I^- = max\{-I, 0\}$. Existence of a solution to this model is provided in the following proposition and proof (see, Federgruen and Katalan [5]).

Proposition 1 *C(L) is minimized by finding the unique minimum of $C_i(L_i)$ for $1 \leq i \leq \frac{N}{2}$, where $C_i(L_i)$ is the long run average cost for item i under basestock policy, L_i.*
$C_i(L_i) = E[h_i((L_i - Q_{2i-1} - Q_{2i})^+) + \pi_i((L_i - Q_{2i-1} - Q_{2i})^-)]$ for $1 \leq i \leq \frac{N}{2}$.
Furthermore, L_i^*=optimal basestock level for item i is obtained by solving a newsvendor problem with $Q_{2i-1} + Q_{2i}$ as the demand distribution.

Proof

1. $\frac{\sum_{i=1}^{\frac{N}{2}} c_i^r}{c}$ is independent of L; therefore, $\frac{\sum_{i=1}^{\frac{N}{2}} c_i^r}{c}$ is constant with respect to L. Hence, $C(L)$ is minimized by minimizing
$\hat{C}(L) = \sum_{i=1}^{\frac{N}{2}} E[h_i((L_i - Q_{2i-1} - Q_{2i})^+) + \pi_i((L_i - Q_{2i-1} - Q_{2i})^-)]$.

2. $\hat{C}(L) = \sum_{i=1}^{\frac{N}{2}} C_i(L_i) = \sum_{i=1}^{N} E[h_i((L_i - Q_{2i-1} - Q_{2i})^+) + \pi_i((L_i - Q_{2i-1} - Q_{2i})^-)]$. Since $h_i()$ and $\pi_i()$ are convex and nondecreasing, $\hat{C}(L)$ is convex and has a unique minimum which may be found by setting $\frac{d\hat{C}(L)}{dQ} = 0$. (Note that $f(g(x))$ is convex if $g(x)$ is convex and f is convex and nondecreasing. In this case $f = h$ and $f = \pi$ are convex and nondecreasing by assumption and $g(x)^+ = L - Q$ and $g(x)^- = Q - L$ can easily be shown to be convex.)

3. Setting $\frac{d\hat{C}(L)}{dQ} = 0$ is equivalent to setting $\frac{d}{dQ_i} C_i(L_i) = 0$, for $1 \leq i \leq \frac{N}{2}$, since $h_i()$ and $\pi_i()$ are non-negative for all i, and
$$\begin{aligned} \frac{d\hat{C}(L)}{dQ} &= \frac{d}{dQ} \sum_{i=1}^{\frac{N}{2}} E[h_i((L_i - Q_{2i-1} - Q_{2i})^+) + \pi_i((L_i - Q_{2i-1} - Q_{2i})^-)] \\ &= \sum_{i=1}^{\frac{N}{2}} \frac{d}{dQ_i} E[h_i((L_i - Q_{2i-1} - Q_{2i})^+) + \pi_i((L_i - Q_{2i-1} - Q_{2i})^-)]. \end{aligned}$$

4. Therefore, $\hat{C}(L)$ is minimized by finding the unique minimum of $C_i(L_i)$ for $1 \leq i \leq \frac{N}{2}$.

□

Note that $C_i(I_i) = E[h_i((L_i - Q_{2i-1} - Q_{2i})^+) + \pi_i((L_i - Q_{2i-1} - Q_{2i})^-)]$ is the newsvendor problem. Newsvendor problems with non-linear overage and underage costs are discussed by Porteus [14]. If costs are linear, i.e., $h(x) = hx$ and $\pi(x) = \pi x$, the problem is to find L_i^* such that $F(L_i^*) = \frac{\pi_i}{\pi_i + h_i}$. (If production is discrete, the closest two integers $\lceil L_i^* \rceil$ and $\lfloor L_i^* \rfloor$, need to be tested, and the integer value which provides the lowest cost selected.)

Proposition 1 may be used to find the optimal basestock levels in the multiproduct system with setups, random yield and backlogging using the results of Grasman [7], which derives the generating function for queue length in a polling model with infinite buffers. Analogous to the queueing representation illustrated in Figure 2, the generating function for queue length represents the generating function for the demand distribution. Using numerical inversion techniques (see, e.g., Abate and Whitt [1]), the generating function may be transformed into the probability distribution required for the newsvendor model.

4 Lost Sales or Expediting

This section addresses setting basestock levels for finite buffer systems, which corresponds to a system with lost sales or expediting shown in Figure 3. We first introduce the cost function that is used, and then discuss the general behavior of the system. Although the analysis does provide service levels (average fraction of lost/expedited customers) and a service level approach could be used, the service level approach would not take into account the significant shortage costs present in this context. Finally, since the calculation of optimal basestock levels is only tractable for small problems, we provide a heuristic for determining the basestock levels along with an example.

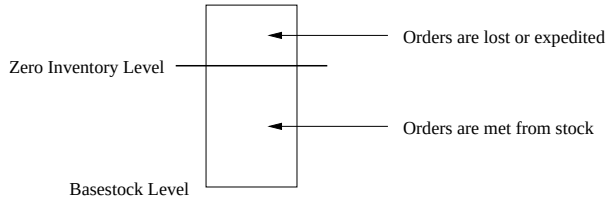

Figure 3: Queueing Representation of Lost Sales or Expediting

4.1 Cost Function

Costs are assigned based on queue length, the number of defects (since defects are no longer produced at a constant rate) and the number of lost arrivals. A holding cost will be charged based on the accumulated inventory level, while rework and lost sales costs will be charged based on the number of defects and lost sales during a cycle. Although setup costs per cycle are constant, setup costs per time is not constant since cycle time increases as the basestock level increase. Thus, we calculate expected costs per minute by dividing by the cycle time. Another result from Grasman et al. [8] will be used to develop a heuristic which determines basestock levels for a production system with lost sales or expediting. Grasman et al. [8] define workload in a queue as the time it would require the server working exclusively on that queue to clear all current customers and cumulative workload as workload summed over all customers and all service periods in the queue during a busy period or vacation period initiated with k customers in a finite buffer polling model with routing. Cumulative workload

is derived and denoted M^k and $\mathcal{M}^k$, for busy periods and vacation periods, respectively.

As discussed earlier, queue length is translated to the inventory level by subtracting the queue length k from the basestock level L. Also, since workload is equal to service times summed over the items in the queue and the service times are independent and identically distributed as well as independent of initial queue length, expected cumulative workload may be translated to expected accumulated inventory by substituting $L-k$ for k in Section 3.2 of Grasman et al. [8], and dividing by the expected service time s. A direct derivation of the expected inventory level is shown in Grasman [7]. Thus the expected accumulated inventory during a busy period, $E[I_{serve}]$, is given by

$$E[I_{serve}] = \sum_{k=0}^{L} \frac{E[M^k]}{s} P(X^p = k),$$

where X^p is the random queue length at a polling instant and s is the expected service time.

Similarly, for vacation periods, the expected accumulated inventory level during a vacation period, $E[I_{vac}]$, is given by

$$E[I_{vac}] = \sum_{k=0}^{L} \frac{E[\mathcal{M}^k]}{s} P(X^v = k),$$

where X^v is the random number of customers in the queue at the start of a vacation and is equal to the number of defects routed to the storage queue during service.

The total inventory of each item accumulated during a production cycle is found by adding the inventory accumulated in the service period and vacation. The total is then multiplied by the linear holding cost per unit of inventory to yield the holding costs per cycle, which is then divided by the cycle time. Thus,

$$\begin{aligned} E[\text{holding costs per cycle}] &= h(E[I_{serve}] + E[I_{vac}]) \text{ and} \\ E[\text{holding cost per minute}] &= \frac{h}{c}(E[I_{serve}] + E[I_{vac}]). \end{aligned}$$

For finite buffers, we include rework costs since no defects are incurred for items that are turned away. Furthermore, increasing the buffer size increases the proportion of time that the server is busy, thus increasing the rate at which defects are produced. The number of defective units of item i, $1 \leq i \leq \frac{N}{2}$, produced in a cycle is equal to the number of customers

routed to the temporary storage queue $2i$. From this fact we can calculate rework costs for item i as

$$\begin{aligned} E[\text{rework cost per cycle}] &= \pi_r^i E[X_{2i}^{2i}] \text{ and} \\ E[\text{rework cost per minute}] &= \frac{\pi_r^i E[X_{2i}^{2i}]}{c}, \end{aligned}$$

where $E[X_{2i}^{2i}]$ is the expected queue length of queue $2i$ when it is polled, which may be calculated recursively from the generating function for queue length derived in Grasman et al. [8], and π_r^i is the linear rework cost for item i.

The expected number of lost units of item $i, 1 \leq i \leq \frac{N}{2}$, in a cycle, $E[L_i^s]$, is equal to the expected number of arrivals in a cycle minus the expected number served in a cycle, $E[N_i]$, where the total number of arrivals in a cycle to real queue $2i-1$ is equal to the number of customers routed from storage queue $2i$ plus the number of external arrivals. Thus, $E[L_i^s] = E[X_{2i}^{2i}] + \lambda_{2i-1}c - E[N_i]$. Shortage costs for item i are calculated from

$$\begin{aligned} E[\text{shortage cost per cycle}] &= \pi^i E[L_i^s] \text{ and} \\ E[\text{shortage cost per minute}] &= \frac{\pi^i E[L_i^s]}{c}, \end{aligned}$$

where π^i is the linear shortage cost for item i.

Expected total cost per minute for item i, $C(L_i)$, is the sum of setup costs, holding costs, rework costs and shortage costs. Thus

$$C(L_i) = \frac{c_i^r}{c} + \frac{h_i}{c}(E[I_{serve}]_i + E[I_{vac}]_i) + \frac{\pi_r^i E[X_{2i}^{2i}]}{c} + \frac{\pi^i E[L_i^s]}{c}, 1 \leq i \leq \frac{N}{2}. \quad (1)$$

4.2 Heuristic

Optimal solutions may be determined by calculating the exact costs of a basestock policy using the procedure described in Grasman et al. [8] and Grasman [7] and partial enumeration. Complete enumeration of policies is not required due to characteristics of the system behavior. However, since finding the optimal solution requires the calculation of the entire set of state probabilities (generating fucntions) for a partial set of basestock policies, it is not tractable to find optimal policies for large problems. Thus, a heuristic, which approximates the state probabilities based on changes in basestock levels, is used to find basestock policies

for finite capacity systems, and then compared to optimal solutions for a set of examples. The heuristic is initialized with a vector of basestock levels, and selects a candidate for expansion based on anticipated shortage costs (probability of being full times shortage cost). The buffer occupancy variables are updated, and the cost effectiveness of the expansion is tested. If the expansion reduces the expected total cost, the next candidate is selected; if not, the heuristic may choose a different queue for expansion or terminate. In detail, the heuristic is as follows.

1. **Initialize Heuristic**

   a) Buffer Occupancy Variables

   For a general vector of basestock levels, $L = (L_1, L_2, ..., L_N)$, the buffer occupancy variables may be calculated in the manner provided in Grasman et al. [8]. A *Mathematica* program has been written for this purpose. For smaller problems, the basestock vector, $L = (L_1 = 1, L_2 = 1, ..., L_N = 1)$, may be calculated quite easily from a set of recursive equations.

   b) Expected Busy Periods

   The expected length of a busy period, which is defined as the duration of service at a queue, for each queue j, $E[B_j]$, may be calculated from the buffer occupancy variables and using

$$E[B_j] \;=\; \sum_{k=0}^{L_j} P(X_j^j = k) E[B_j^k], \tag{2}$$

   where $E[B_j^k]$ is the expected busy period in queue j starting with k customers, and may be calculated from a set of recursive equations.

   c) Expected Number Served in a Busy Period

   The expected number served in a busy period for each queue j, $E[N_j]$, may be directly calculated from

$$E[N_j] \;=\; \frac{E[B_j]}{s_j}, \tag{3}$$

where s_j is the expected service time for queue j.

d) Cycle Time and Vacation Periods

Expected cycle time, c, and expected vacation periods, $E[V_j]$ are calculated from

$$\begin{aligned} c &= r + \sum_{j=1}^{N} E[B_j] \\ E[V_j] &= c - E[B_j], \end{aligned} \tag{4}$$

where r = total expected setup time in a cycle.

e) Lost Sales (Shortage)

Recall that the number of lost units in a cycle, $E[L_i^s]$ is equal to the total number of arrivals in a cycle minus the number served in a cycle. Thus, the expected number of lost arrivals at a real queue j is

$$E[L_j^s] = E[X_{j+1}^{j+1}] + \lambda_j c - E[N_j]. \tag{5}$$

f) Average Cost

Calculate total average cost using Equation (1).

2. **Find Candidate Queue for Capacity Expansion**

Select a candidate queue i, let $L_i = L_i + 1$ and leave all other basestock levels unchanged.

a) External Arrivals During Vacation

Since the number of external arrivals during a vacation period is according to a Poisson process, the probability of j external arrivals to queue i, h_j^{ext}, is

$$h_j^{ext} = E\left[\frac{(\lambda_i V_i)^j e^{-\lambda_i V_i}}{j}\right]. \tag{6}$$

b) Buffer Occupancy Variables Under Candidate Capacity Increase

We denote the probabilities under the candidate capacity increase by h'_j. We then set $h'_j = P(X_i^i = j)$ for $j < L_i$, and calculate h'_{L_i} from

$$h'_{L_i} = \sum_{k=0}^{L_i} h_k^{ext} h_{j-k}^{int}, \tag{7}$$

where h_j^{int} is the probability of j internal (routed) arrivals to queue i.

c) Probability of Needing Additional Capacity

Under the candidate increase, the probability of needing additional buffer space is

$$h'_{L_i+1} = 1 - \sum_{k=0}^{L_i} h'_k \tag{8}$$

d) Approximate Expected Moments

The expected moments of buffer occupancy, busy period, number served and lost sales under the candidate capacity increase are denoted by $E[X_i^i]'$, $E[B_i]'$, $E[N_i]'$, $E[X_{i+1}^{i+1}]'$ and $E[L_i^s]'$, respectively. Since we are not considering the impact of the basestock level increase on the entire system, we may only approximate the values of $E[X_i^i]$ and $E[B_i]$. Thus we know that

$$\begin{aligned} E[X_i^i]' &\geq \sum_{k=0}^{L_i+1} k h'_k, \\ E[B_i]' &\geq \sum_{k=0}^{L_i+1} h_k E[B^k]. \end{aligned} \tag{9}$$

From these values we may make the following approximations:

$$\begin{aligned} E[N_i]' &\approx \frac{E[B_i]'}{s_i}, \\ E[L_i^s]' &\approx \lambda_i c - E[N_i] y_i, \\ E[X_{i+1}^{i+1}]' &\approx E[N_i]'(1 - y_i). \end{aligned} \tag{10}$$

The first two approximations come from Equations (3) and (4), while the last approximation comes from the fact that the number of routed customers from queue i to queue $i+1$ is binomially distributed with parameters N_i and $1 - y_i$.

e) Approximate Cost

Calculate total approximate average cost using Equation (1).

Repeat Step 2 for all real queues.

f) Select Queue to Increase Capacity

Increase capacity by one at the queue that has the lowest cost when its buffer size is increased. Note that even if the cost does not decrease from the previous iteration, it is only an approximate cost, so we must complete the next iteration of the heuristic.

For the chosen item i, set $L_i = L_i + 1$. We denote the values from the last iteration by the superscript *old*, and the update values with the superscript *new*. Also, let n be the index of the queue that is presently being updated. Since increasing the basestock level (buffer size) at queue i stochastically increases the queue length at the start of service, we must update the buffer occupancy variables by setting

$$h_j^{new} = h_j'.$$

Also, since increasing the basestock level at queue i increases the busy period at queue i, we must update the busy period by setting

$$E[B_i]^{new} = E[B_i]'.$$

Finally we calculate $\Delta E[B_i] = E[B_i]^{new} - E[B_i]^{old}$, and let $n = (i+2)\ mod\ N$.

3. **Calculate Updated Values (Busy Periods)**

Increasing the busy period at a queue i increases the cycle time and the vacation period at all other queues. Thus the buffer occupancy variables and busy periods are stochastically increased and must be updated.

a) Additional External Arrivals

First, we calculate the number of additional arrivals due to the increase in busy period. Let $P_j^n(t)$ = probability of j arrivals to queue n during time t. Then

$$P_j^n(t) \quad = \quad \frac{(\lambda_n t)^j e^{-\lambda_n t}}{j}, \tag{11}$$

where $t = \sum \Delta E[B_i]$.

When t is small, the effects on the other queues are minimal; thus, there is no need to update the buffer occupancy variables or busy period. Therefore, we choose one of the following:

- IF $P_0^n(t) > 1 - \epsilon_1$ THEN let $n = (n+2)$ *mod* N and GO TO Step 3a).
- IF $P_0^n(t) > 1 - \epsilon_1$ and all queues have been *sufficiently* updated THEN GO TO Step 4.
- IF $P_0^n(t) > 1 - \epsilon_1$ and no busy periods were updated THEN GO TO Step 5.
- IF $P_0^n(t) < 1 - \epsilon_1$ THEN continue to Step 3b),

where ϵ_1 is the allowable error; i.e., when the probability of additional arrivals is very small the heuristic continues.

b) Update Buffer Occupancy Variables

The buffer occupancy variables are equal to the buffer occupancy variables under the previous system plus the number of external arrivals during the additional busy periods plus the additional number of internal arrivals caused by defects. In Step 4 we consider the routed defects, but first, in this step we update the buffer occupancy variables by letting

$$h_j^{new} = \sum_{k=0}^{j} P_k^n(t) h_{j-k}^{old}. \tag{12}$$

c) Update Busy Periods

Using the update buffer occupancy variables, we update the busy periods using

$$E[B_n]^{new} = \sum_{k=0}^{L_n} h_k^{new} E[B^k]. \tag{13}$$

d) Let $n = (n+2)$ *mod* N GO TO Step 3a).

4. **Calculate Updated Values (Number Served and Routed)**

Increasing the busy period at a queue increases the number served in the busy period thus increasing the number of defects produced and the number of defects routed to the temporary storage queue, which in turn increases the number of internal arrivals at the real queue, thus increasing the buffer occupancy variables.

Let $n = i$

a) Update Number Served

We may calculate the number served and change in number served by

$$\begin{aligned} E[N_n]^{new} &= \frac{E[B_n]^{new}}{s_n} \text{ and} \\ \Delta E[N_n] &= E[N_n]^{new} - E[N_n]^{old}. \end{aligned} \tag{14}$$

b) Bound on $\Delta E[X_{n+1}^{n+1}]$

We bound the expected number of routed defects from queue n using the fact that the number of routed defects is binomially distributed with parameters N_i and $1 - y_n$, and noting that some defects may be lost (where at this point we are ignoring losses at storage queues). Thus,

$$\Delta E[X_{n+1}^{n+1}] \leq \Delta E[N_n](1 - y_n). \tag{15}$$

When $E[X_{n+1}^{n+1}]$ is small, the effects on the other queues are minimal; thus, there is no need to update the buffer occupancy variables or busy period. Therefore, we choose one of the following:

- IF $\Delta E[X_{n+1}^{n+1}] < \epsilon_2$ THEN let $n = (n+2)\ mod\ N$ GO TO Step 4a).
- IF all real queues have been updated THEN let $n = i$ GO TO Step 3.
- IF $\Delta E[X_{n+1}^{n+1}] > \epsilon_2$ continue to Step 4c),

where ϵ_2 is the allowable error; i.e., when the expected number of additional routed customers is very small, the heuristic continues.

c) Update Number Routed

Recursive formulas may be used to calculate $P(N_n = k)$, and then using the binomial distribution,

$$P(X_{n+1}^{n+1} = x) = \sum_{x=0}^{k} \binom{k}{x} (1 - y_n)^x y_n^{k-x} P(N_n = k). \tag{16}$$

d) Update Buffer Occupancy Variables

The buffer occupancy variables at queue n are updated using

$$h_j^{new} = \sum_{k=0}^{j} h_k^{ext} h_{j-k}^{int}, \tag{17}$$

where h_k^{ext} = probability of k Poisson arrivals during the updated vacation period and $h_j^{int} = P(X_{n+1}^{n+1} = j)$.

e) Update Busy Period

The busy period is then updated using

$$E[B_n]^{new} = \sum_{k=0}^{L_n} h_k^{new} E[B^k]. \tag{18}$$

Choose one of the following:

- IF all real queues have been updated THEN let $n = i$ GO TO Step 3
- ELSE let $n = (n + 2)$ *mod* N GO TO Step 4a).

5. **Evaluate Cost Effectiveness**

   Calculate costs using Equation (1)

   Choose one of the following:

   - IF cost decreases THEN accept change GO TO Step 2.
   - IF cost increases THEN reject change, select next queue from Step 2 GO TO Step 3
   - IF all candidates have been evaluated THEN STOP and GO TO Step 6.

6. **Find Solution**

   Use the current basestock levels as the heuristic policy. For small examples, the resulting solution may be compared to the actual costs of the policy using the method provided in Grasman et al. [8] or may be compared to simulation results.

4.3 Numerical Examples

Optimal solutions are determined by calculating the exact costs of a basestock policy using the procedure described in Grasman et al. [7] and Grasman [8]and partial enumeration. Complete enumeration of policies is not required due to characteristics of the system behavior. However, since finding the optimal solution requires the calculation of the entire set of

state probabilities for a partial set of basestock policies, it is not tractable to find optimal policies for large problems. Thus, the heuristic, which approximates the state probabilities is used to find basestock policies and then compared to optimal solutions for a set of examples.

Detailed examples of the heuristic may be found in Grasman [8]. A battery of 3-item examples is provided to further demonstrate the performance of the heuristic, especially the effects of setup times and yield rates. In order to insolate these effects, the arrival rates and service rates are kept constant with $\lambda_i = .75$ and $s_i = .25$. Traffic intensity is defined as the effective utilization of the system including routed arrivals due to defects, i.e., $\rho = \sum_i \rho_i = sum_i \frac{\lambda_i s_i}{y_i}$. The heuristic found the optimal solution in all cases, and Table 1 shows the results of the the test.

Table 1. Numerical Examples

Example	y_1	y_2	y_3	r_1	r_2	r_3	% Error
1	0.5	0.5	.75	.25	.25	.5	0.88%
2	0.5	0.5	.75	.25	.5	.5	0.43%
3	0.5	0.5	.75	.5	.5	.5	2.27%
4	0.5	0.75	.5	.25	.25	.25	2.12%
5	0.5	0.75	.75	.25	.25	.5	1.51%
6	0.5	0.75	.75	.25	.5	.5	2.14%
7	0.5	0.75	.75	.5	.5	.5	2.27%
8	0.75	0.5	.5	.25	.25	.25	1.65%
9	0.75	0.5	.75	.25	.25	.5	1.26%
10	0.75	0.5	.75	.25	.5	.5	2.29%
11	0.75	0.5	.75	.5	.5	.5	1.39%
12	0.75	0.75	.5	.25	.25	.25	0.91%
13	0.75	0.5	.75	.5	.5	.5	1.38%

The numerical results illustrate the effectiveness of the heuristic as the optimal basestock levels were obtained in each of the cases. In addition, the heuristic cost was within 2.50% of the optimal cost for all cases, thus the approximation of the total cost for a system with lost sales is quite accurate.

5 Conclusions and Future Work

Utilizing previous results, this paper sets basestock levels in the multiproduct system with setups and random yields. For a system with backlogging, we introduce the cost function and show that it is minimized by solving N single item newsvendor problems. We then derived the optimal basestock level for arbitrary convex and nondecreasing holding and shortage costs, and demonstrated that the solution simplifies to the standard newsvendor solution when holding and shortage costs are linear. For a system with lost sales or expediting, we introduced the cost function and provided a heuristic for finding the basestock levels. The heuristic was then demonstrated via an example.

Future research will continue to explore this problem by addressing limitations of the current formulation including model parameters and policy structure. The basestock policy dictates that the service discipline at each queue is exhaustive. By eliminating this assumption, other disciplines such as k-limited, gated, and fractional service could be used to optimize a more general service policy. Since cyclic polling also limits the flexibility of the system, more general periodic polling as well as random polling structures should be incorporated; also, non-Poisson demand structures, including batch and other dependent demand should be considered.

Related formulations within the current basestock framework, include altered arrival rate or service rate and direct (instant) feedback into the queue. Arrival and service rates may be altered to take into account the actual rate in which nondefective items are produced. Under these alterations, the need for routing would be eliminated and would thus reduce the computational complexity of the formulation. Although the distribution of various variables would change, the first moments would remain the same. Further, if immediate inspection was allowed, demand caused by defects would be immediately known and thus satisfied during the current cycle.

Acknowledgments

This research was supported by National Science Foundation Grant DMI-9713727.

References

[1] Abate, J., and W. Whitt, "Numerical Inversion of Probability Generating Functions," *Operations Research Letters*(12), 245-251, 1992

[2] Bohn, R.E., and C. Terwiesch, "The Economics of Yield Driven Processes," *Working Paper*, The Wharton School, University of Pennsylvania.

[3] Bollapragada, S., and T.E. Morton, "Myopic Heuristics For the Random Yield Problem," *Operations Research*(47)5, 713-722, 1999.

[4] Elmaghraby, S.E., "The Economic Lot Scheduling Problem (ELSP): Review and Extensions," *Management Science*(24)6, 587-598, 1978.

[5] Federgruen, A., and Z. Katalan, "The Stochastic Economic Lot Scheduling Problem: Cyclic Basestock Policies with Idle Times," *Management Science*(42)6, 783-796, 1996.

[6] Gilbert, S., and H. Emmons, "Managing a Deteriorating Process in a Batch-Production Environment," *IIE Transactions*(27)2, 233-243, 1995.

[7] Grasman, S.E., "Production Strategies for Random Yield Processes," *Ph.D. Dissertation*. University of Michigan, 2000.

[8] Grasman, S.E., T.L. Olsen and J.R. Birge, "Finite Buffer Polling Models with Routing," *Under Review: European Journal of Operational Research*, 2002.

[9] Henig, M., and Y. Gerchak, "The Structure of Periodic Review Policies in the Presence of Random Yield," *Operations Research*(38)4, 634-643, 1990.

[10] Jung, W.Y., and C.K. Un, "Analysis of Finite-Buffer Polling System with Exhaustive Service Based on Virtual Buffering," *IEEE Transactions on Communications*(42)12, 3144-3149, 1994.

[11] Krieg, G.N., and H. Kuhn , "A Decomposition Method for Multi-Product Kanban Systems with Setup Times and Lost Sales," *IIE Transactions* (34)7, 613-625, 2002.

[12] Krieg, G.N., and H. Kuhn, "Analysis of Multi-Product Kanban Systems with State-Dependent Setups and Lost Sales," *Annals of Operations Research*, to appear.

[13] Krieg, G.N., and H. Kuhn, "Production Planning of Multi-Product Kanban Systems with Significant Setup Times," in *Operations Research Proceedings 2000*, B. Fleischmann, R. Lasch, U. Derigs, W. Domschke, and U. Rieder, editors. Springer, 2001.

[14] Porteus, E.L., "Stochastic Inventory Theory," in *Handbooks in Operations Research and Management Science Volume 2: Stochastic Models*, D.P. Heyman and M.J. Sobel , editors. North-Holland, 1990.

[15] Rajaram, K., and U.S. Karmarkar, "Product Cycling with Uncertain Yiels: Analysis and Application to the Process Industry," *Operations Research* (50)4, 680-691, 2002.

[16] Sidi, M., H. Levy and S.W. Fuhrmann, "A Queueing Network with a Single Cyclically Roving Server," *Queueing Systems*(11), 121-144, 1992.

[17] Yano, C.A., and H.L. Lee, "Lot Sizing with Random Yields: A Review," *Operations Research*(43)2, 311-334, 1995.