Efficient Algorithms for Buffer Space Allocation

Stanley B. Gershwin [a,*] and James E. Schor [b]

[a] *Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, Massachusetts 02139-4307*

E-mail: gershwin@mit.edu

[b] *Analytics, Inc., 101 Rogers Street, Cambridge, Massachusetts 02142*

E-mail: jschor@analytics-consulting.com

This paper describes efficient algorithms for determining how buffer space should be allocated in a flow line. We analyze two problems: a *primal* problem, which minimizes total buffer space subject to a production rate constraint; and a *dual* problem, which maximizes production rate subject to a total buffer space constraint. The dual problem is solved by means of a gradient method, and the primal problem is solved using the dual solution. Numerical results are presented. Profit optimization problems are natural generalizations of the primal and dual problems, and we show how they can be solved using essentially the same algorithms.

1. Introduction

1.1. Problem description

Production systems are often organized with machines or work centers connected in series and separated by buffers. This arrangement is often called a *flow line*, or *transfer line*, or *production line*. A five-machine line is represented in Figure 1, in which the squares represent machines and the circles represent buffers. Material moves in the direction of the arrows, from upstream inventory to the first machine for an operation, to the first buffer where it waits for the second machine, to the second machine, etc.

Material flow may be disrupted by machine failures or variable processing times. Buffers are inserted between machines to limit the propagation of disruptions, and this increases the average production rate of the line. Inclusion of buffers requires additional capital investment and floor space, which may be

* Corresponding author.

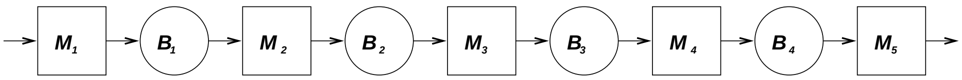

Figure 1. Flow line

expensive. Buffering also increases in-process inventory. If the buffers are too large, the work-in-process inventory and capital costs incurred will outweigh the benefit of increased productivity. If the buffers are too small, the machines will be underutilized or demand will not be met.

Often, the limitation on the amount of in-process inventory is not due to physical constraints. Instead this limitation is a control policy, for example, a version of the kanban policy (Muckstadt and Tayur, 1995a). It is essential to determine buffer sizes to achieve the desired performance.

This paper describes efficient algorithms for buffer space allocation. Much literature analyzes the effects of machine unreliability and buffer sizes on system performance. Less effort has been expended to develop methods for optimizing or improving the allocation of buffer space. This paper is a summary of Schor (1995), in which can be found all details of algorithms and numerical results.

1.2. Approach and Outline

This paper provides a unified view of solutions to several versions of the problem. They are organized according to problem type and model type.

- **Problem type**
  We describe algorithms which choose buffers sizes for a flow line. They are based on the qualitative properties of the production rate described in Section 2. The number of machines and the reliability and speed parameters of each machine are assumed to be specified.
  - ∗ **Primal** The goal of the *primal* is to minimize the total buffer space required for the line to meet or exceed a given average production rate. This formulation is appropriate if either floor space or the buffering mechanism is expensive, if work-in-process inventory is inexpensive, and if an average production rate is mandated.
  - ∗ **Dual** The goal of the *dual* is to maximize the production rate achievable with a given total buffer space. This is appropriate in cases where the total floor space is fixed, where the number of buffer locations is fixed, and where the problem is how to space the machines to get the maximum benefit from

the buffers. In addition, the dual is used in this paper as a subproblem in solving the primal.

∗ **Constrained profit maximization** The goal is to maximize profit subject to a constraint on total buffer space. The solution technique is similar to that of the dual problem.

∗ **Unconstrained profit maximization** The buffer space constraint is relaxed. The solution technique is similar to that of the primal problem. It uses the constrained profit maximization problem in the same way that the primal algorithm uses the dual.

- **Model type** We consider systems that may be represented by discrete or continuous material flow models. In both, the processing time is deterministic. The discrete material model has the advantage of better representing the discrete nature of typical factories, but it suffers from two important drawbacks: it is restricted to systems with equal processing times, and the buffer sizes (which are the decision variables in the allocation problems) are integers. The continuous model is better suited to optimization because its variables are real numbers; it can be used for discrete part factory design by rounding the buffer sizes after the optimization has been completed.

  ∗ **Discrete material** In the discrete material, deterministic processing time system, the machines have equal processing times and geometrically distributed failure and repair times. The probability of Machine M_i failing during an operation time while it is operating (operational and not starved or blocked) is p_i. The probability of a repair during an operation time while it is under repair is r_i. The size of Buffer B_i is N_i. N_i is a nonnegative integer; p_i and r_i are real numbers in [0,1]. See (Buzacott, 1967; Gershwin, 1994). We use an approximate decomposition method, developed by Gershwin (1987), to determine production rate and average buffer levels. The DDX algorithm (Dallery, David, and Xie, 1988), solves the decomposition equations efficiently and produces estimates for production rate and average buffer levels.

  ∗ **Continuous material** The machines have different processing rates and exponentially distributed failure and repair times. The amount of material produced during $(t, t + \delta t)$ by Machine M_i while it is fully operational (operational, not starved or blocked, and not slowed down by an adjacent machine) is $\mu_i \delta t$. The probability of Machine M_i failing during such an interval is $p_i \delta t$. The probability of a repair during $(t, t + \delta t)$ while it is under repair is $r_i \delta t$. The size of Buffer B_i is N_i. μ_i, p_i, r_i, and N_i are nonnegative real numbers. The decomposition for the continuous model, and the ADDX algorithm, with which we evaluate its performance measures, were developed by Burman (1995). The ADDX algorithm approximately determines the production rate and average in-process inventory for this kind of system, by a decomposition method which is an extension of the Gershwin (1987)–Dallery-David-Xie (1988) method.

Numerical results are provided for selected cases.

Qualitative properties of flow line models are discussed in Section 2. The primal and dual problem formulations are presented in Section 3. The primal and dual problem solution methods are summarized in Sections 4 and 5. The behavior of the algorithms is discussed in Section 6. We reformulate the problems to maximize profit (and therefore to include inventory costs) in Section 7. We conclude in Section 8.

1.3. Prior Literature

There is a substantial literature on the analysis of transfer or flow lines (Dallery and Gershwin, 1992). This literature is mainly concerned with the prediction of performance. Much of it is aimed at evaluating the production rate of a system with specified machines and buffers. One ultimate goal of the literature is to aid the designers of such systems to make efficient choices of machines and buffers. To do this, two elements are needed: a fast, accurate method of evaluating systems, and an efficient way of selecting systems to evaluate. Fast evaluation methods exist, but as Martin (1994) says, "... the more practical goal of optimizing system design is one less visibly pursued in the literature."

Here, we do not review the literature on evaluation; we concentrate instead on flow line optimization papers and on papers about relevant qualitative properties of production lines. The main contribution of this paper is a set of algorithms that efficiently select the minimal buffer space in a flow line to achieve a specified production rate. The algorithms are based on analytical approximations of the Buzacott model of a flow line, so they are appropriate for automated systems, and they are fast. By contrast, some papers in the literature are based on simulation; others use different analytic models; and nearly all have other objective functions.

1.3.1. Qualitative Properties

In order for gradient methods to work well, certain qualitative properties are needed. Although the independent variables often are integers, there should be some kind of *continuity*: a small change in a buffer size should lead to a small change in the system's performance. (Many line optimization papers assume this implicitly.) In addition, many methods rely on *monotonicity*: an increase in a buffer's size (while all the other buffer sizes are increased or held constant) increases production rate. Finally, for a gradient method to find a unique optimum, it is helpful for some form of *concavity* to hold. That is, the increase in production rate due to a unit increase in buffer size decreases as the buffer size increases. (It is interesting to observe how much of the work on line optimization was done *before* the publication of the papers described in this section.) Combinatorial approaches, which do not exploit these properties, can be expected to be relatively inefficient.

Several papers have studied qualitative properties of models of manufacturing systems that are relevant to optimization. The properties have been proved for classes of systems other than those we study here, but we believe that the systems are similar enough to justify assuming the properties we need. Numerical experiments support this assumption.

Shanthikumar and Yao (1989) studied cyclic queuing networks with finite buffers. Service processes were exponential, with rates that are increasing functions of the number of customers in the queue. They showed that the production rate is an increasing function of the buffer sizes. Adan and Van der Wal (1989) proved monotonicity of the production rate with respect to buffer sizes of acyclic systems using a sample path method. That is, they also showed that the production rate increases as each buffer is enlarged. They prove this for a more general class of networks than the one studied here.

Meester and Shanthikumar (1990) studied tandem queuing systems with reliable exponential servers and finite storage areas. They showed that the production rate is an increasing, concave function of the buffer sizes. Anantharam and Tsoucas (1990) studied systems of queues in series with single exponential servers, finite buffers, and communication blocking (blocking before service). They showed that the throughput (production rate) is a concave function of the buffer sizes. Glasserman and Yao (1996) studied similar systems under a wide variety of blocking mechanisms, and demonstrated concavity for systems with exponential processing times, and other qualitative properties for general service time distributions. Dallery, Liu, and Towsley (1994) generalized the results of Shanthikumar and Yao (1989) and Meester and Shanthikumar (1990) by establishing the concavity of throughput as a function of buffer sizes (and initial conditions, for a closed network) for a greatly extended class of distributions of service times. Useful properties for a general class of related systems were studied by Tayur (1992) and Muckstadt and Tayur (1995a). Buzacott and Shanthikumar (1992) summarized much of the relevant literature on manufacturing system design.

1.3.2. Simulation-based approaches

Many papers have proposed design methodologies based on evaluation by simulation. Because simulation takes longer than analytical methods on the same problem, even the most recent of such methods are slower than analytical methods. The main advantage of simulation is that it has the potential for dealing with a larger class of systems than analytical methods.

Barten (1962) simulated a set of cases, and interpolated a curve fit for delay time as a function of storage capacity for several line lengths, assuming normally distributed operation times. He formed a cost function, and calculated the optimal storage capacity. Freeman (1964) simulated short lines (two and three stages) with unreliable machines. Even under these limited conditions, he was able to show that the placement and sizes of buffers is important, and he provided some preliminary rules about how to allocate storage space.

Anderson and Moodie (1969) simulated two classes of production lines with reliable machines: one class had identical, normally distributed service times; the other had identical, exponentially distributed service times. All the buffers in each line had identical sizes. In each case, average in-process inventory and *delay* (evidently 1.0-utilization) were fitted to simple functions of the length of the lines and the buffer sizes. Cost functions were also developed, and optimal buffer sizes were calculated from them. Martin (1993, 1994) extended this work by refining the delay and inventory functions and generalizing the cost and profit functions. He studied the performance of the line as a function of line length and buffer size.

Powell (1994) and Powell and Pyke (1994) simulated flow lines to determine the optimal locations and amounts of buffering. In their systems, operation times are random but machines or workstations do not fail. As a consequence, the buffer sizes they found were small. They also suggested some rules of thumb for buffer sizing and placement.

Chow (1987) and Liu and Lin (1994) used simulation to construct functions to predict the throughput and the CV of the interdeparture time of a reliable two-machine line. Liu and Lin (1994) improved Chow's approach by designing a procedure that required fewer runs to construct these functions. They also proposed an aggregation method, which represented the long line as a single two-machine line, to extend their results to longer lines. They used this in a dynamic programming formulation to solve the production rate maximization problem. They solve the minimum total buffer space problem by using the solution to the production rate maximization problem. Starting with buffers of size zero, they evaluated the maximum production rate for each value of total space. They increased the space by one and repeated until the throughput constraint was satisfied. They performed experiments on lines of less than ten machines with a total buffer space of less than fifty. When evaluating longer lines they only considered bottleneck machines.

Another important class of simulation-based papers optimize lines by means of *perturbation analysis*. Ho, Eyler, and Chien (1979) was the first of these. In this paper, a Buzacott-like model of a transfer line (one with discrete parts, identical constant processing times, and geometrically distributed repair and failure times) was simulated. By means of a sophisticated analysis, an estimate of the gradient of the production rate with respect to all the buffer sizes was determined from only one run of the simulation. This estimate was then used in a steepest descent method for solving the dual problem. (See also Ho, Eyler, and Chien, 1983). Caramanis (1987) applied a similar technique to a line with deterministic but different processing times (and exponential repair and failure times). The cost function depended on buffer capacity, average inventory, and production rate. A more recent paper, which studied the optimization of production rate in a continuous material model of a flow line (subject to linear constraints on the line parameters) by similar techniques is Plambeck, Fu, Robinson, and Suri (1993).

1.3.3. Exploration of the optimum by exact numerical evaluation

Hillier and So (1991) studied systems of unreliable machines and finite buffers. They represented the systems as finite queues in series with two-stage Coxian distributed operation times. Using an exact numerical method for short lines (5 stations or less) and small buffers (4 or less), they evaluated all plausible optimal distributions of buffer sizes in order to find the optimum. Hillier, So, and Boling (1993) studied a problem which is similar to the dual described here, the maximization of the production rate subject to a total buffer space constraint. As is typical of the papers in the series by Hillier and his co-authors, they evaluated a large number of cases to find the optimum, and characterized the optimal distribution. This methodology requires small, simple lines: their machines were reliable with exponential processing times, their lines had no more than eight stations, and their buffers were no larger than 14. When all machines were the same, they observed an *inverted bowl phenomenon*, in which the optimal distribution has smaller buffers near the ends of the line, and larger buffers in the interior.

The main goal of these papers has been to develop an understanding of the structure of optimal systems. By contrast, our goal is to develop efficient methods for obtaining an optimum (or a near-optimum). In addition, these papers tend to deal with models of reliable work centers with processing times (which may be appropriate for manual systems). This paper deals with deterministic processing time models which are appropriate for automated systems with fixed processing times and random failures.

1.3.4. Other algorithms and approaches

Early papers The early papers on buffer allocation in flow lines made various simplifications or restrictions in order to obtain results. Small systems are easier to analyze than large systems, and reliable machines with exponential processing times are easier to analyze than machines with more realistic behavior.

One of the earliest papers in the production line literature was that of Sevast'yanov (1962). This paper included an exact solution of a two-machine line, and an approximate analysis of longer lines. (Some ideas that have been implemented in the last fifteen years were anticipated by this paper.) It also contained formulations of storage size optimization problems that are similar to those here. Numerical results were not included, no doubt due to the unavailability of computers, but theoretical results demonstrated that buffer sizes should be chosen so that the apparent upstream failure rate should equal the apparent downstream failure rate, for an observer in each buffer.

Hatcher (1969) studied reliable systems with exponential processing times and finite buffers, and investigated the allocation of capacity in three-stage, two-buffer systems. (However, there is an error in the evaluation of the line. See Knott, 1970.) Kraemer and Love (1970) optimize the profit in a two-stage, one-buffer system with reliable, exponential machines. Profit is revenue minus inventory carrying cost minus storage facility cost; revenue is proportional to production rate; inventory carrying cost is proportional to expected in-process inventory; and storage facility cost is proportional to the buffer size.

Sheskin (1976) studied the allocation of buffer space in systems like ours: those with unreliable machines with equal, deterministic processing times, and finite buffers. However, he restricted his analysis to the case where $r_i + p_i = 1$ (in our notation), which is equivalent to assuming that the machine repair state at time $t+1$ is independent of the state at time t. In addition, he assumed time-dependent failures, i.e., that machines could fail even if they were starved or blocked. A decomposition method was used to produce numerical results for small systems with small buffers. These results led to some rules of thumb on the allocation of buffer space to maximize production rate. Soyster, Schmidt, and Rohrer (1979) used the same model to study the maximization of production rate subject to general linear inequality constraints on buffer sizes. They approximated the production rate for small systems and used an integer programming package to find optimal allocations of buffer space.

Altiok and Stidham (1983) studied systems with exponentially distributed service, repair, and failure times. Their goal was to maximize average profit, which increased with production rate and decreased with total average inventory. There were no constraints on total buffer space. They evaluated the steady state distribution by numerically solving the transition equations, so they were limited to small systems.

More recent papers Chow (1987) developed an aggregation procedure for evaluating production lines. He developed a dynamic programming procedure to maximize production rate subject to a total buffer space constraint.

Smith and Daskalaki (1988) developed a method for general networks of reliable exponential machines with random routing and finite buffers. They formed an unconstrained optimization problem. In their profit objective, production rate had a positive coefficient, and average buffer level had a negative coefficient. A standard optimization method was used. By contrast, we deal with unreliable machines with constant processing time and fixed routing in a tandem network, and we develop a special purpose optimization method.

Jafari and Shanthikumar (1989) studied a problem similar to our dual: maximizing the production rate subject to the total storage space constraint. They included scrapping in their model, and they evaluated their lines in a manner similar to the method used here. They studied two optimization methods: a dynamic programming algorithm, and a greedy heuristic.

Park (1993) developed a two-phase tree search branch and bound technique to solve some of the same problems that we investigate here. He restricted his analysis to the same discrete material model that we study. He identified four kinds of problems: Problem 1 is the same as our primal; Problem 2 is a generalization of our dual (in which the buffer sizes may be weighted with different coefficients, and in which there may be more than one linear constraint); Problem 3 seeks to minimize the total buffer space subject to the production rate requirement of Problem 1 (and our primal) and to the generalized buffer limits of Problem 2. Problem 4 seems to have two objectives: to simultaneously maximize production rate while minimizing total buffer space. Park also included a survey of earlier buffer allocation work.

Jacobs and Meerkov (1993, 1994) studied the *improvability* of production line systems. While their definition of "improvable" appears equivalent to "non-optimal," their goal was not the design of systems. Rather, they sought methods for efficiently improving the performance of existing systems using data that is available as the systems operate. Jacobs and Meerkov (1993, 1994) employ the concept of *improvability* to determine how buffer space should be allocated. They propose a method for determining whether the individual buffer sizes may be changed, while holding the total buffer space constant, so that the production rate will increase. The flow line model they study is similar to that of Sheskin (1976) and Soyster, Schmidt, and Rohrer (1979). Kuo, Lim and Meerkov (1994) described an application.

Seong, Chang, and Hong (1994a) studied our dual problem: they maximize the production rate subject to a constraint on the total buffer size. (This is our dual, and Park's Problem 2.) In Seong, Chang, and Hong (1994b), they studied the same objective function as Altiok and Stidham (1983) with general linear constraints on buffer sizes. Material was assumed continuous, machine operation speeds were deterministic and equal, and repairs and failures were exponentially distributed. They solved the problem with a gradient projection algorithm. They used a similar approximation of the gradient of the production rate as we do below. Seong, Chang and Hong (1994a) used a gradient method to solve the production rate maximization problem for a flow line with exponentially distributed failure, repair, and processing times. Seong, Chang, and Hong (1994b) also solved this problem and the profit maximization problem for a specified total buffer space for a continuous flow line. We compare our results with theirs in Section 7. Dogan and Altiok (1995) use decomposition to estimate the throughput gradient vector with respect to repair rates. Their ultimate goal is to optimize "repair rate allocation". Altiok and Yamashita (1997) observed a similar primal/dual relationship as ours, and solved the dual problem using a dynamic programming approach.

Gershwin and Goldis (1995) employed a gradient method to solve the primal problem. Their algorithm was based on the observation that if the production rate is expanded to first order the problem may be formulated as an integer linear program. They guaranteed that their solutions are optimal or near-optimal. The algorithm presented here converges much more quickly in the majority of the cases we study.

2. Definitions and Properties

2.1. Models, Parameters, and Notation

In the following, N_i, the size of Buffer B_i, is the decision variable, and P is the production rate of the line. The line has k machines and $k-1$ buffers. In the discrete material formulation, all machines are assumed to have the same operation time, which we use as the time unit. The parameters of Machine M_i are p_i, the probability of a failure during a time unit while the machine is operating; and r_i, the probability of a repair during a time unit while the machine is down.

In the continuous material formulation, workpieces are treated like a continuous fluid. This is one way of modeling machines that have different speeds. The rate at which Machine M_i processes material, when it is operational and neither starved nor blocked is μ_i. That is, it processes $\mu_i \delta t$ units of material in a time interval of length δt. The reliability parameters of Machine M_i are p_i and r_i. They are probability rates: $p_i \delta t$ is the probability of a failure during a time period of length δt during which the machine is operating and neither starved nor blocked; and $r_i \delta t$ is the probability of a repair during a δt time period during which the machine is down. Details of both models can be found in Gershwin (1994).

Although the production rate P is a function of machine speeds and reliabilities (and, in general, other attributes not treated here), we vary only buffer sizes, so we write $P = P(N_1, \ldots, N_{k-1})$.

2.2. Machines

Isolated quantities are quantities of a machine that are independent of the rest of the line. The *isolated efficiency* of Machine M_i is given by

$$e_i = \frac{r_i}{r_i + p_i}.$$

In the discrete material model, the operation time is 1 and the production rate is the average number of parts produced in one time unit. Therefore, the *isolated production rate* is the same as the isolated efficiency.

In the continuous material model, the production rate of Machine M_i is μ_i when it is operating and not influenced by other machines or buffers. Therefore, its isolated production rate is given by

$$\rho_i = \mu_i e_i.$$

2.3. Lines

2.3.1. Quantitative Properties

The production rate P when all buffer sizes are infinite is the minimum of the isolated production rates of all the machines in the line. That is, in the discrete material case,

$$P(\infty, ..., \infty) = \min_{i=1,...,k} e_i$$

In the continuous material case,

$$P(\infty, ..., \infty) = \min_{i=1,...,k} \rho_i$$

In the deterministic processing time model, the production rate when all buffers have size 0 is (Buzacott 1967),

$$P(0, ..., 0) = \frac{1}{1 + \sum_{i=1}^{k} \frac{p_i}{r_i}} \tag{1}$$

A similar equation is available for the continuous material case. It is more complex when the μ_i are not the same.

When there are more than two machines and all N_i are neither infinite nor zero, the production rate and average inventory levels cannot be calculated analytically or exactly even numerically. Decomposition methods (Gershwin,

Machine	r_i	p_i
1	0.35	0.037
2	0.15	0.015
3	0.4	0.02

Table 1
Three-machine line parameters

1987; Gershwin, 1994; Burman, 1995) and iterative algorithms (Dallery, David, and Xie, 1988; Gershwin, 1994; Burman, 1995) must be used. These algorithms work by calculating $r_u(i)$, $p_u(i)$, $r_d(i)$, and $p_d(i)$ (and also $\mu_u(i)$ and $\mu_d(i)$ for the continuous material case), the parameters of fictitious machines that represent the behavior of the flow of material into and out of Buffer B_i. As indicated in Section 5.4, these methods should be adapted for use in the algorithms described here.

2.3.2. Qualitative Properties

We assume the following qualitative properties:

- **Continuity** A small change in any N_i creates a small change in P.
- **Monotonicity** The production rate increases monotonically in each N_i.
- **Concavity** The production rate appears to be a concave function of the vector $(N_1, \ldots, N_{k-1})$.

These properties are evident in Figure 2, which shows how production rate varies with buffer sizes for the three-machine line whose parameters are given in Table 1. Figure 3 shows the curves of constant P for this case. (The *optimal curve* is described in Section 5.2.) The literature provides proofs of these properties for similar systems.

3. Primal and Dual Problem

In all of these problems, the buffer sizes have lower bounds. In the discrete material formulation, it is convenient to restrict N_i to be greater than or equal to 4. This is because the functional forms of the production rate and average buffer levels are different for $N_i < 4$ than for $N_i \geq 4$. We therefore write $N_i \geq N^{\mathsf{MIN}}$, where $N^{\mathsf{MIN}} = 4$ in the discrete material case and $N^{\mathsf{MIN}} = 0$ in the continuous material case.

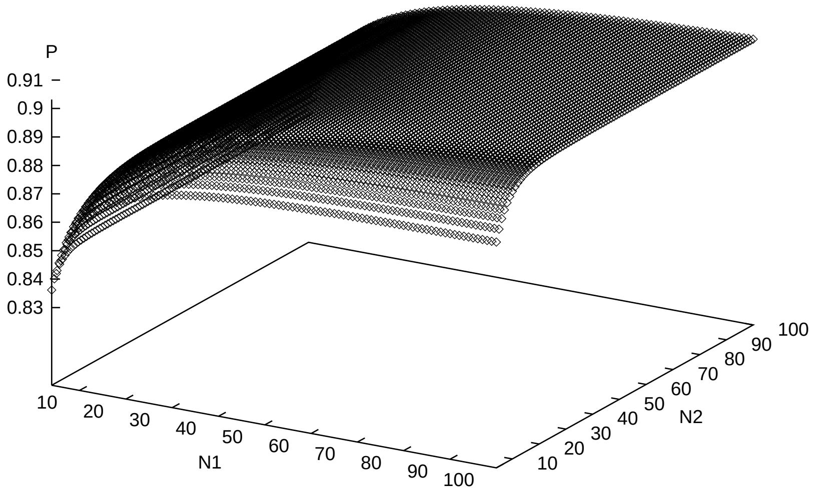

Figure 2. P vs. N_1 and N_2

3.1. Primal Problem

In the primal problem, we seek the buffer sizes $(N_1, ..., N_{k-1})$ that minimize total buffer space N^{TOTAL} such that the production rate P is greater than or equal to a specified value P^*. Or,

$$\text{Minimize } N^{\mathsf{TOTAL}} = \sum_{i=1}^{k-1} N_i \tag{2}$$

subject to

$$P(N_1, \ldots, N_{k-1}) \geq P^*; \ P^* \text{ specified} \tag{3}$$

$$N_i \geq N^{\mathsf{MIN}}, i = 1, ..., k-1$$

This problem is difficult because the constraint $P \geq P^*$ cannot be expressed in closed form. Consequently, even if we know a value of $(N_1, \ldots, N_{k-1})$ satisfying $P(N_1, \ldots, N_{k-1}) = P^*$, it is difficult to construct another $(N_1, \ldots, N_{k-1})$ that will also be on that surface.

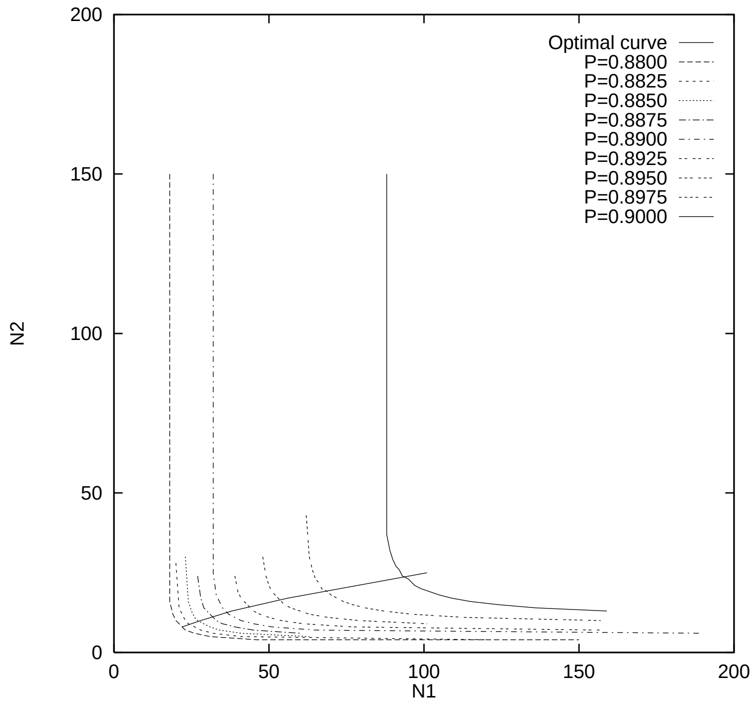

Figure 3. Iso-P lines

The qualitative properties ensure that a solution will be found with constraint (3) satisfied with equality. Since the properties appear to hold strictly, there will be a unique solution.

3.2. Dual problem

In the dual problem, we seek the buffer sizes $\boldsymbol{N} = (N_1, ..., N_{k-1})$ that maximize the production rate P such that the total buffer space is equal to a specified value N^{TOTAL}. That is,

$$\text{Maximize} \;\; P(N_1, \ldots, N_{k-1}) \tag{4}$$

subject to

$$N^{\mathsf{TOTAL}} = \sum_{i=1}^{k-1} N_i; \;\; N^{\mathsf{TOTAL}} \, \text{specified} \tag{5}$$

$$N_i \geq N^{\mathsf{MIN}}, i = 1, ..., k-1$$

The input to this problem is N^{TOTAL} and the outputs are the optimal P_{MAX} and $(N_1, \ldots, N_{k-1})$. The resulting maximum value of P is written $P_{\mathsf{MAX}}(N^{\mathsf{TOTAL}})$.

4. Dual Algorithm

The dual algorithm is a gradient algorithm. An initial guess $\boldsymbol{N}$ is selected and a direction to move in $(N_1, ...N_{k-1})$ space is determined. A linear search is then conducted in that direction until a maximum is encountered. This becomes the next guess. A new direction is chosen and the process continues until no improvement is realized.

The dual problem is relatively easy to solve, and the dual algorithm is relatively easy to implement, because the constraint set (5) is a plane. As a result, large steps may be taken.

To determine the search direction, we find the gradient $\boldsymbol{g}$ according to a forward or backward difference formula. The forward difference is

$$g_i = \frac{P(N_1, \ldots, N_i + \delta N, \ldots, N_{k-1}) - P(N_1, \ldots, N_i, \ldots, N_{k-1})}{\delta N} \tag{6}$$

Monotonicity guarantees that $g_i \geq 0$ for all i.

We must select $\delta \boldsymbol{N} = \hat{\boldsymbol{N}} - \boldsymbol{N}$ (where $\boldsymbol{N}$ is the most recent guess of the dual solution and $\hat{\boldsymbol{N}}$ is the next guess) to achieve the greatest increase in P and to continue to satisfy constraint (5). We construct the search direction $\boldsymbol{p}$ by projecting $\boldsymbol{g}$ onto the hyperplane $\sum_{i=1}^{k-1} \delta N_i = 0$ as follows:

$$\bar{g} = \frac{1}{k-1} \sum_{i=1}^{k-1} g_i; \quad p_i = g_i - \bar{g} \tag{7}$$

An exception occurs when $N_q = N^{\mathsf{MIN}}$ and $p_q \leq 0$ for some q. We calculate the new direction by deleting the q_{th} component of $\boldsymbol{g}$ and setting $p_q = 0$. Then we recalculate the other components of $\boldsymbol{p}$ using (7), and the new $\boldsymbol{g}$. This process is continued until a feasible step may be taken or all components of $\boldsymbol{g}$ are deleted.

We find the scalar A such that $\hat{\boldsymbol{N}} = \boldsymbol{N} + \delta \boldsymbol{N} = \boldsymbol{N} + A\boldsymbol{p}$ maximizes P. Then we calculate the next gradient $\boldsymbol{g}$ and repeat. This process ends when all components of $\boldsymbol{g}$ are sufficiently small.

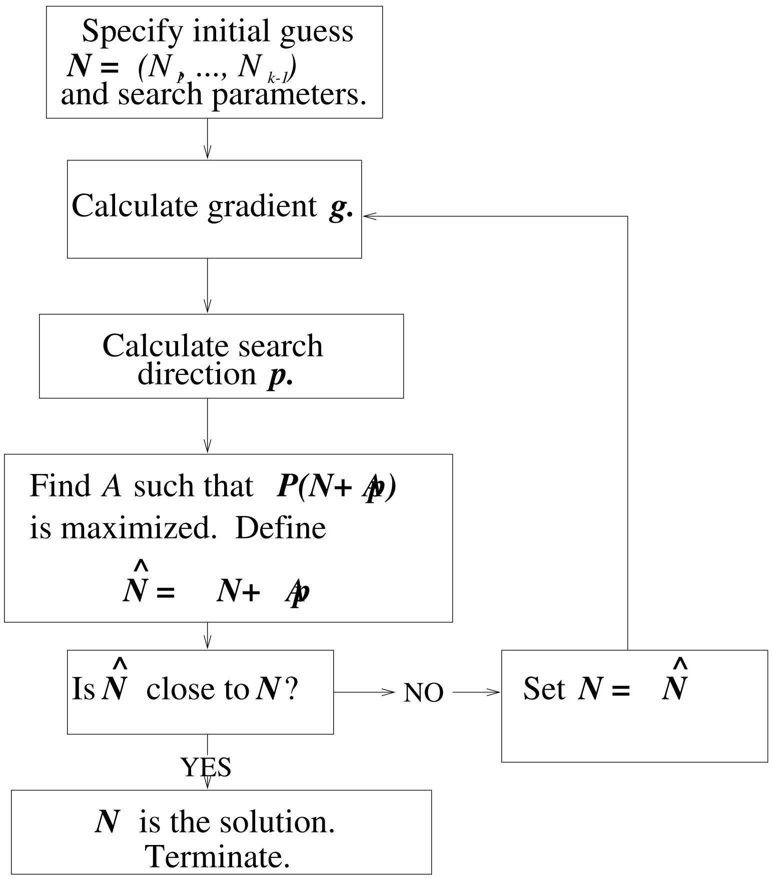

Figure 4. Block Diagram of Dual Algorithm

A block diagram of this algorithm appears in Figure 4.

5. Primal Algorithm

5.1. Algorithm Structure

It is difficult to construct $(N_1, ..., N_{k-1})$ to satisfy constraint (3) with equality. By using the dual algorithm as part of the solution of the primal problem, we avoid having to search surfaces like those of Figure 2.

We solve the primal problem by dividing it into two subproblems: the dual and the one-dimensional primal problem (ODPP). The primal problem is: find N^{TOTAL} such that $P_{\mathsf{MAX}}(N^{\mathsf{TOTAL}}) = P^*$. The dual algorithm is used to evaluate $P_{\mathsf{MAX}}(N^{\mathsf{TOTAL}})$.

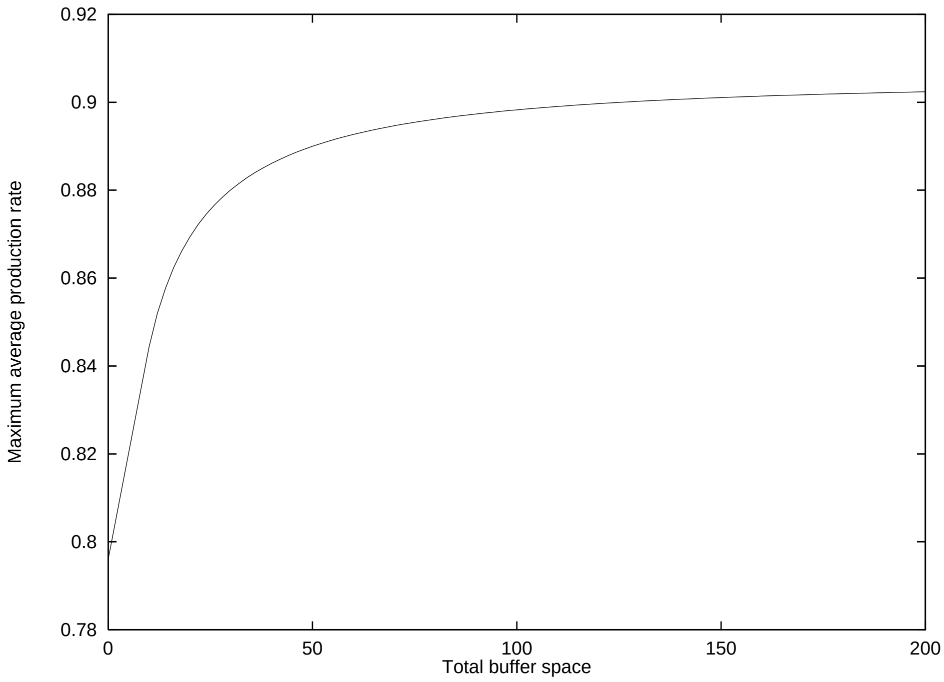

Figure 5. P_{MAX} vs. N^{TOTAL}

Let $\boldsymbol{N}$ (N^{TOTAL}) be the distribution of buffer space that solves the dual problem with total buffer space N^{TOTAL}. The *optimal curve* is the set of $\boldsymbol{N}$ (N^{TOTAL}) as N^{TOTAL} varies over a range of values. An example of an optimal curve appears in Figure 3.

The nearly straight optimal curve is helpful in making the algorithm efficient. Linearizing it leads to a very good initial guess of $(N_1, ..., N_{k-1})$ for the dual algorithm.

5.2. One-Dimensional Primal Problem (ODPP)

Figure 5 is the graph of $P_{\mathsf{MAX}}(N^{\mathsf{TOTAL}})$ for the three-machine, two-buffer system determined in Table 1. (This is the same system that generated Figure 3.)

Consider the following one-dimensional problem:

$$\text{Minimize } \; N^{\mathsf{TOTAL}} \tag{8}$$

subject to

$$P_{\mathsf{MAX}}(N^{\mathsf{TOTAL}}) \geq P^*;\ P^* \text{ specified}$$

$$N^{\mathsf{TOTAL}} \geq (k-1)N^{\mathsf{MIN}}$$

Let the solution be called $N^{\mathsf{TOTAL}*}$. This is essentially the same as the primal problem because the same $N^{\mathsf{TOTAL}*}$ satisfies both, and because $\boldsymbol{N}$ $(N^{\mathsf{TOTAL}*})$ satisfies the primal.

Schor (1995) develops a one-dimensional search which is based on the property that $P_{\mathsf{MAX}}(N^{\mathsf{TOTAL}})$ is monotonically increasing in N^{TOTAL}. The algorithm requires an initial guess, N^1. We evaluate $P_{\mathsf{MAX}}(N^1)$ using the dual algorithm. We also need $P_{\mathsf{MAX}}(0)$, the production rate of the line when all buffers are of size zero. The evaluation of $P_{\mathsf{MAX}}(0)$ is described in Section 5.4 under "Necessity." We construct a linear approximation of $P_{\mathsf{MAX}}(N^{\mathsf{TOTAL}})$, and we use this to estimate N^2, the second guess.

For $j \geq 2$, $P_{\mathsf{MAX}}(N^j)$ is evaluated using the dual algorithm, j is incremented and then we iterate. Each iteration uses N^j and N^{j-1} to construct the next linear approximation of $P_{\mathsf{MAX}}(N^{\mathsf{TOTAL}})$. The process continues until $P_{\mathsf{MAX}}(N^{\mathsf{TOTAL}})$ is sufficiently close to P^*.

5.3. Algorithm

A block diagram of this algorithm appears in Figure 6. The algorithm is initialized with $N^0 = (k-1)N^{\mathsf{MIN}}$, $j = 2$, and with values of N^1 and search parameters specified. $P_{\mathsf{MAX}}(N^1)$ is obtained by solving the dual; $P_{\mathsf{MAX}}(N^0)$ is obtained from Buzacott's zero-buffer formula in the discrete material case (in which case it is an approximation), or from evaluating the ADDX algorithm for $N_i = 0$ for all i in the continuous case.

N^j is determined from N^{j-2} and N^{j-1} by approximating the inverse function of $P(N^{\mathsf{TOTAL}})$ with a straight line. That is, we approximate

$$N^{\mathsf{TOTAL}} \approx mP_{\mathsf{MAX}} + b$$

where

$$m = \frac{N^{j-1} - N^{j-2}}{P_{\mathsf{MAX}}(N^{j-1}) - P_{\mathsf{MAX}}(N^{j-2})}$$

$$b = N^{j-1} - mP_{\mathsf{MAX}}(N^{j-1})$$

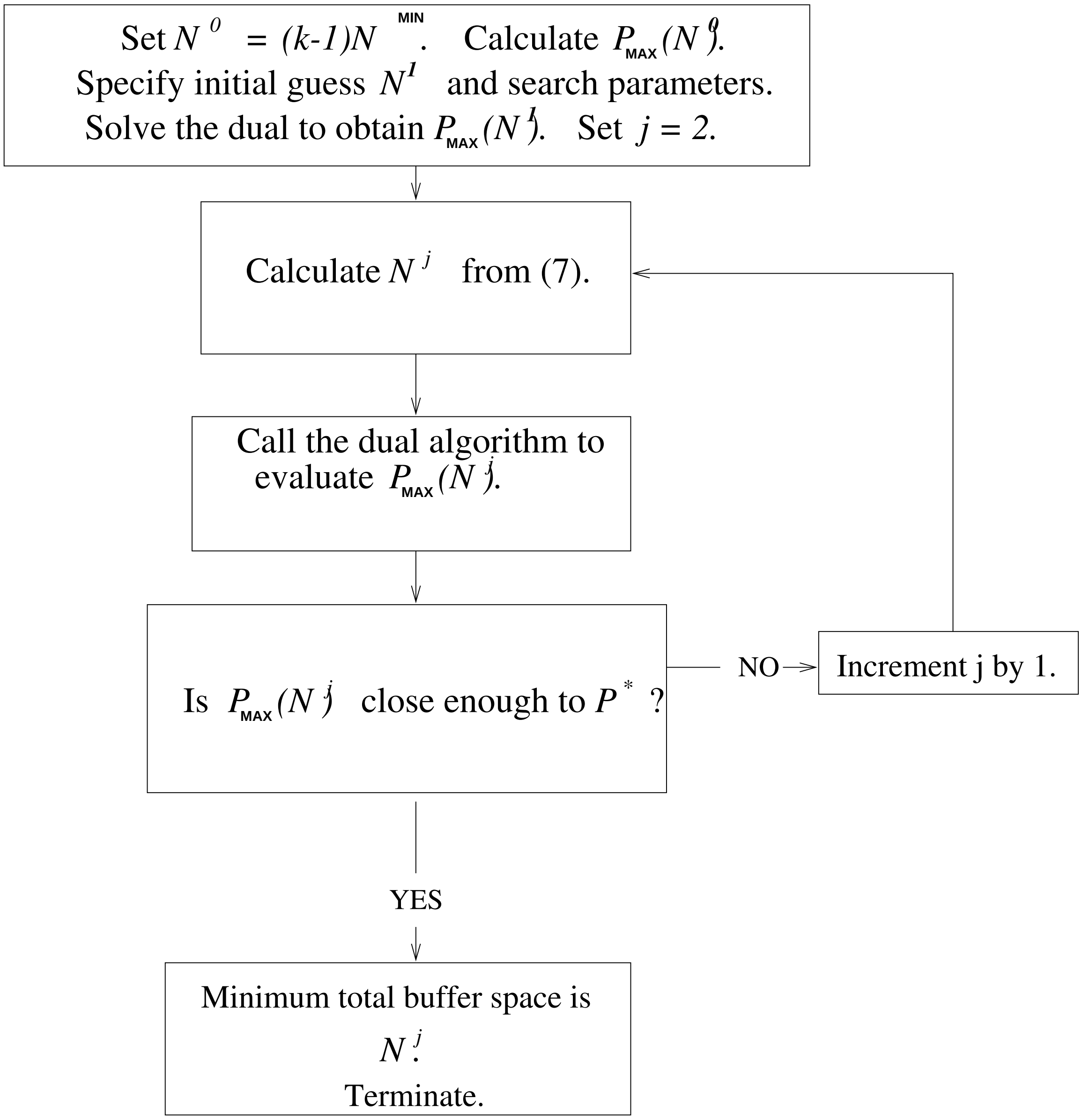

Figure 6. Block Diagram of Primal Algorithm

and we select

$$N^j = mP^* + b.$$

We can also write this as

$$N^j = AN^{j-1} + BN^{j-2} \tag{9}$$

where

$$A = \frac{P^* - P_{\mathsf{MAX}}(N^{j-2})}{P_{\mathsf{MAX}}(N^{j-1}) - P_{\mathsf{MAX}}(N^{j-2})}$$

$$B = \frac{P_{\mathsf{MAX}}(N^{j-1}) - P^*}{P_{\mathsf{MAX}}(N^{j-1}) - P_{\mathsf{MAX}}(N^{j-2})}$$

This approach must be modified when N^{TOTAL} is large because the P_{MAX} vs. N^{TOTAL} curve (Figure 5) is very flat.

The dual algorithm is used to evaluate $P_{\mathsf{MAX}}(N^j)$. If $P_{\mathsf{MAX}}(N^j)$ is sufficiently close to P^*, the algorithm terminates.

5.4. Implementation issues

There are some important issues to consider when implementing these algorithms.

Feasibility We check feasibility by requiring that the isolated production rate of each machine is greater than the production rate target. That is, in the discrete material case, a solution exists only if

$$P^* < \min_i \frac{r_i}{r_i + p_i}$$

and in the continuous material case, a solution exists only if

$$P^* < \min_i \frac{r_i \mu_i}{r_i + p_i}$$

Necessity We also want to determine if there is a need for buffers. If the production rate of the line with no buffers is greater than P^* there is no reason to run the algorithm. When the deterministic processing time model is used, calculate $P(0, ...0)$ according to (1). If the processing rates of the machines are different, we invoke the ADDX algorithm with $N_i = 0$ for all i to evaluate $P(0, ...0)$. In either case, if $P(0, ...0) \geq P^*$, then all the optimal buffer sizes are 0.

ADDX and DDX algorithm initialization These are iterative algorithms that must be initialized whenever they are invoked. Here, instead of a standard initialization, we use the final value of $(N_1, ..., N_{k-1})$ from the last evaluation. This decreases the number of iterations required for the algorithms to converge. (See Gershwin and Goldis (1995).)

Dual initialization Just as the next guess for N^{TOTAL} is obtained from locally linearizing the $P_{\mathsf{MAX}}(N^{\mathsf{TOTAL}})$ graph, a good guess for $(N_1, ..., N_{k-1})$ can be obtained from linearizing the optimal curve.

Rounding In the final step of the algorithms for continuous material models (if the actual line is discrete), we round N^i so that the solution is integer. This can sometimes reduce P to below the target P^*, so some adjustment is made.

6. Algorithm Behavior

6.1. Dual

Schor (1995) observes that the solutions are either optimal or very close to optimal in all of the cases studied. The accuracy was established by comparing results with an algorithm of Gershwin and Goldis (1995). Gershwin and Goldis established the accuracy of their solutions by comparing them to the results of an exhaustive search.

When a continuous model is used for a discrete system, Schor observed that the accuracy of the solution is affected by the rounding methodology.

6.1.1. Discrete Five Machine System

The optimal allocation algorithm of Ho, Eyler, and Chien (1979) is an iterative simulation-based algorithm. It has an efficient method that determines the production rate and gradient by simulating a transfer line once rather than k times. In this algorithm, A is determined empirically (and is evidently constant throughout the algorithm), and the next guess $\hat{\boldsymbol{N}}$ is $\boldsymbol{N} + A\,\boldsymbol{p}$, where $\boldsymbol{N}$ is the previous guess. Simulation is then used to estimate $P(N_1, \ldots, N_{k-1})$ and $\boldsymbol{g}$ for each guess. When no improvement is realized, the algorithm stops.

In this example, we are determining the optimal buffer allocation for the five-machine discrete material line described in Table 2. The final buffer distributions are presented in Table 3. The production rates for both buffer distributions are calculated using the DDX algorithm and simulation. The distribution of buffer space seemed to be quite different, but the total buffer space and the production rate were very close.

Machine	1	2	3	4	5
MTTR $= 1/r$	11	19	12	7	7
MTTF $= 1/p$	20	167	22	22	26

Table 2
Five-machine line (Ho *et al.* 1983)

The simulation was run for 50 runs of 100,000 cycles each and it required more than 5 minutes. To run the dual algorithm required 0.3 seconds. (This experiment was performed on a Sun SparcStation 2.)

	Buffer i						
Case	1	2	3	4	$\boldsymbol{N}$	P (DDX)	P (Simulation)
Ho *et al.*	5	11	8	7	31	0.4914	0.4931
Dual	7	10	10	4	31	0.4943	0.4962

Table 3
Buffer allocation for a five-machine line

6.1.2. Continuous Seven Machine System

In this example, the machines have different processing times so we cannot use the discrete material analytical model. Ho et al. have no such restriction because they use simulation. As a result, we compare the optimal distribution that we get from using the continuous material model and then rounding with the distribution obtained by Ho et al.

We are seeking the optimal buffer allocation for the seven-machine line described in Table 4. The final buffer distributions are presented in Table 5. The production rates reported in the last column of Table 5 are calculated using the ADDX algorithm.

6.2. Primal

The deterministic processing time model is used for all the examples in this chapter. We have initialized $N_i = 5 \;\; \forall \; i$ for all examples unless we explicitly state otherwise.

Again, Schor (1995) compared his results with those of Gershwin and Goldis (1995), who established the accuracy of their solutions by comparison with the

Machine	1	2	3	4	5	6	7
MTTR = $1/r$	450	760	460	270	270	650	320
MTTF = $1/p$	820	5700	870	830	970	1900	1100
Cycle time = $1/\mu$	40	34	39	38	37	40	43

Table 4

Seven-machine line (Ho *et al.* 1979)

			Buffer i					
Case	1	2	3	4	5	6	$\boldsymbol{N}$	ADDX production rate
Ho *et al.*	5	11	8	7	19	4	54	0.0126
Dual	8	10	13	10	9	4	54	0.0128

Table 5

Buffer allocation for a seven-machine line

results of an exhaustive search. The search was performed for several five-machine lines and buffer sizes of up to 20. In all cases, N^{TOTAL} is either the same or it differs by 1.

6.2.1. Comparison with prior work

In this section, we compare the performance of the primal algorithm to algorithms developed by Park (1993) and Gershwin and Goldis (1995). Park's algorithm uses a two-phase tree search method and does not guarantee an optimal solution. Gershwin and Goldis' "combined" algorithm (called GG here) is a gradient algorithm which takes advantage of the decomposition to obtain a crude and cheap approximation of the gradient in the early iterations, and uses a more accurate approximation at the end. (This could be done with any iterative algorithm.)

Both the Park and GG algorithms solve the primal problem for the discrete material, deterministic processing time model. Both algorithms use the DDX algorithm to evaluate $P(N_1, \ldots, N_{k-1})$. Both papers present results for the twelve-machine line described in Table 6.

Table 7 compares the total buffer size, production rates, and number of long line and two-machine evaluations for experiments with two different values of P^*. The initial guess for the primal and GG algorithms, in both cases, is $\boldsymbol{N} = (5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5)$. (Park's algorithm does not require an initial

i	1	2	3	4	5	6	7	8	9	10	11	12
r_i	0.35	0.15	0.4	0.4	0.3	0.2	0.3	0.3	0.4	0.4	0.3	0.25
p_i	0.037	0.015	0.02	0.03	0.03	0.01	0.02	0.02	0.02	0.03	0.03	0.01

Table 6

Twelve-machine line (Park 1993)

		Park			GG			Primal		
Case	Target rate	Actual rate	Long line evals.	$\boldsymbol{N}$	Actual rate	Evals. long line (two-mach.)	$\boldsymbol{N}$	Actual rate	Evals. long line (two-mach.)	$\boldsymbol{N}$
1	0.85	0.8396	739	84						
2	0.85	0.8420	1,107	84						
3	0.85	0.8505	1,659	93	0.8507	182	87	0.8507	114	87
4	0.85	0.8505	1,838	93		(125,923)			(79,140)	
5	0.895	0.8950	510	390	0.8950	1,173 (1,481,928)	242	0.8950	342 (534,820)	243

Table 7

Comparison of algorithms

guess.) Cases 1-4 are all the same problem: Park runs his algorithm with different search parameters.

We feel that the number of two-machine evaluations is the best measure of the run time of the algorithms because this measure is computer-independent. Park reports the number of long line evaluations. This is also independent of the computer, but it is a less absolute measure because there is more flexibility in the long line evaluation. (The initial conditions and the solution accuracy can vary during the course of the algorithm.)

For cases 1-4 the GG algorithm and the primal algorithm achieve identical solutions but the primal algorithm requires about one third fewer two-machine evaluations. Park's algorithm does not satisfy the production rate constraint in cases 1 or 2. In the cases where Park does satisfy the constraint, a larger $\boldsymbol{N}$ is calculated. When Park's algorithm satisfies the production rate constraint, it requires substantially more long line evaluations than the others.

For $P^* = 0.895$, the solution of the primal algorithm has one more buffer space than the GG algorithm. The primal algorithm requires the least computational effort in all cases.

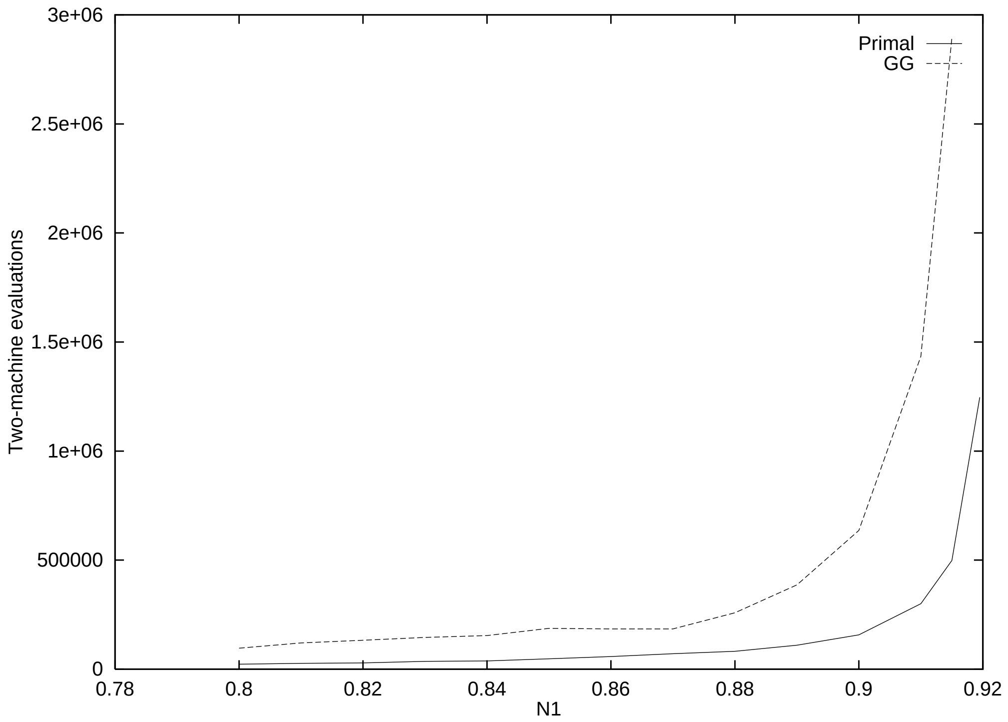

Figure 7. Two-machine evaluations as a function of P

Increasing P^* Figure 7 illustrates how execution time is impacted by P^* for the primal and GG algorithms. The number of two-machine evaluations, for the line described in Table 8, is reported. When P^* approaches the infinite buffer production rate (0.92) the number of two-machine evaluations required increases significantly. The primal and GG algorithms calculate the same value for $\boldsymbol{N}$ in all cases.

Long lines Table 9 demonstrates how increasing the length of the line impacts computational effort. Each line is composed of ten-machine blocks. Each block is described in Table 8. $P^* = 0.88$ for all the lines. The significant increase in computational effort is a consequence of three factors. The first is the increased number of two-machine evaluations required for the DDX algorithm to converge. The second is that each gradient calculation requires an additional long line evaluation for each added machine. The third is that as the line increases the total buffer space increases and the algorithm must search a larger space.

Machine	i 1	2	3	4	5	6	7	8	9	10
r_i	.094	.095	.045	.078	.069	.094	.095	.045	.078	.069
p_i	.007	.008	.003	.004	.006	.007	.008	.003	.004	.006
P_i	.93	.92	.94	.95	.92	.93	.92	.94	.95	.92

Table 8

Ten-machine block

Number of machines	Two-machine evals. Primal	GG	$\boldsymbol{N}$ Primal	GG
10	82,176	258,661	433	443
20	1,814,868	6,867,878	995	995
30	15,955,968	47,157,058	1,556	1,557

Table 9

Long line computation times

6.2.2. Effect of a bottleneck

This example comes from Gershwin and Goldis (1995). Consider the design of a 20-machine production line. The machines have already been selected, and the only decision remaining is the amount of space to allocate for in-process inventory. The common operation time is one operation per minute. The target production rate is .88 parts per minute. The goal is to determine the smallest amount of in-process inventory space so that the line meets that target.

Three versions of this line are analyzed.

- In Case 1, each machine is unreliable with mean time to fail (MTTF) 200 minutes and mean time to repair (MTTR) 10.5 minutes. Every machine could therefore operate in isolation at rate .95 parts per minute on the average, and easily meet the target. However, if there were no in-process inventory allowed, each machine's failure would cause every other machine to be forced idle, and the actual production rate would only be .487. To achieve the target production rate, the minimal required inventory space is room for 430 parts, distributed according to the figure.
- Case 2 and 3 are similar to Case 1, except that Machine 5 (and only Machine 5) is replaced by a less reliable model. There is thus a well-defined bottleneck. See Figure 8. In both cases, the reliability of Machine 5 is reduced to .905.

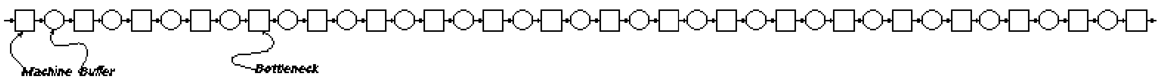

Figure 8. Twenty-Machine Line

Line	Production rate with no buffers	Minimal buffer space for production rate target (.88 parts/min)
Case 1	.487	430
Case 2	.475	485
Case 3	.475	523

Table 10
Summary of Twenty-Machine Line Results

* In Case 2, MTTF = 100 and MTTR = 10.5 minutes.
* In Case 3, MTTF = 200 and MTTR = 21 minutes.

The results are summarized in Table 10 and Figure 9. With no buffering between the machines, the production rate of both Cases 2 and 3 would be .475. Again, each machine (including the bottleneck) could meet the required production rate in isolation, but the complete system cannot without inventory buffering between the machines. The minimal required total inventory space is room for 485 parts for Case 2 and 523 for Case 3. Figure 9 displays how the space is distributed in each case. The zero-buffer production rates were calculated from Buzacott (1968).

Observations:

- The additional inventory space is allocated approximately symmetrically around the bottleneck, but it is not all placed precisely at the bottleneck.
- Case 3 needs more space than Case 2. This suggests that increased repair time is worse than increased failure frequency.
- Case 1 exhibits the inverted bowl phenomenon, observed by Hillier, So, and Boling (1993). A line with identical machines requires less buffer space at the ends because of the usual assumption that the first machine is never starved and the last machine is never blocked.

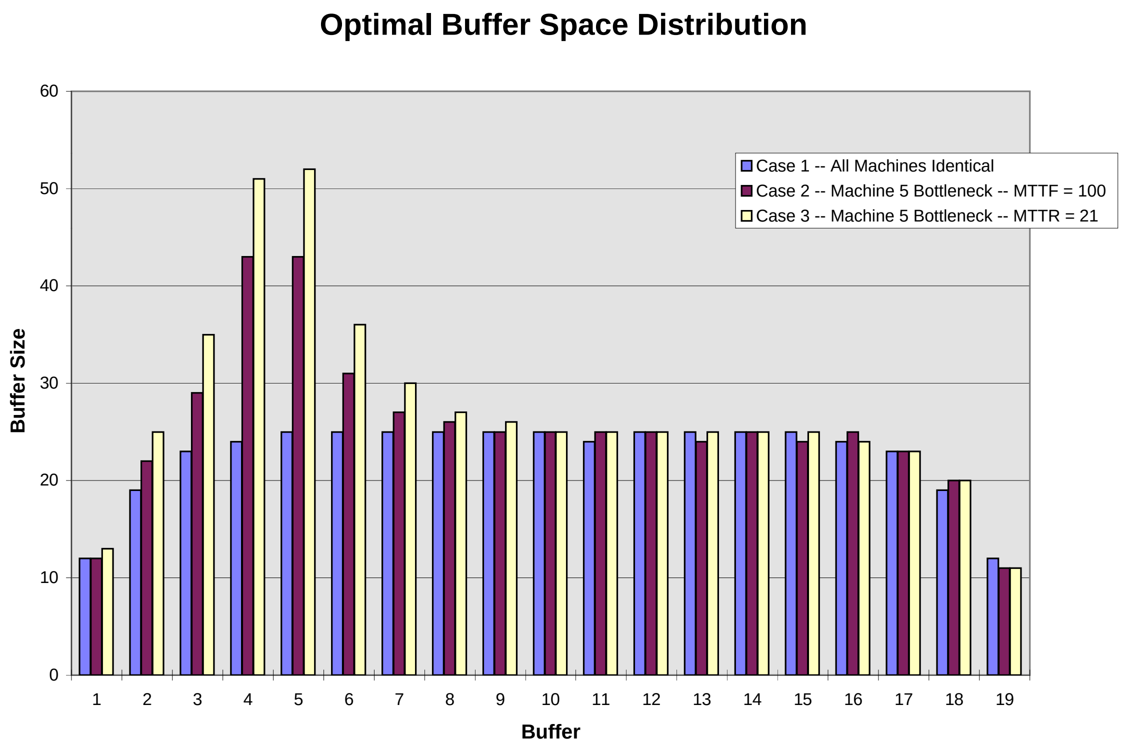

Figure 9. Optimal Buffer Distributions

- The graphs are somewhat uneven. This is due mainly to the fact that we have restricted the solution to integers: we are assuming that the line is producing discrete items. In addition, the results shown may be slightly non-optimal.

7. Profit Maximization

Here, we generalize the objective function to include inventory holding costs. Let $\bar{n}_i$ be the average amount of inventory in Buffer B_i. This quantity is an output of the DDX and ADDX algorithms. Let Φ, a_i, and b_i be non-negative cost coefficients. The profit is

$$\Pi = \Phi P(N_1, \ldots, N_{k-1}) - \sum_{i=1}^{k-1} a_i N_i - \sum_{i=1}^{k-1} b_i \bar{n}_i$$

in which Φ is the profit per item sold, when all costs other than inventory costs at this system are considered. The cost of inventory space is linear in the size of the space, and the cost per unit space in Buffer B_i is a_i. The cost of inventory held is linear in the average amount of inventory, and the cost per item stored in Buffer B_i is b_i.

We know of no published papers that discuss the qualitative properties of profit as a function of buffer sizes (comparable to Section 2.3.2). Gershwin (1994) shows that average buffer level may be either a concave or convex function of buffer size for a two-machine line. The shape of this function depends upon the reliability parameters and processing rates of the machines. Consequently, if profit is a function of average buffer level, it may not be concave.

7.1. *Profit Maximization With Total Buffer Space Constraint*

$$\text{Maximize } \Pi \tag{10}$$

subject to

$$N^{\mathsf{TOTAL}} = \sum_{i=1}^{k-1} N_i; \quad N_i \geq N^{\mathsf{MIN}}$$

7.2. *Profit Maximization Without Space Constraints*

$$\text{Maximize } \Pi \tag{11}$$

subject to

$$N_i \geq N^{\mathsf{MIN}}$$

7.3. *Properties of* Π

We can no longer rely on the objective function being monotonic or concave. However, numerical experience suggests that it has a single maximum, and this appears to be enough for the algorithms to work effectively.

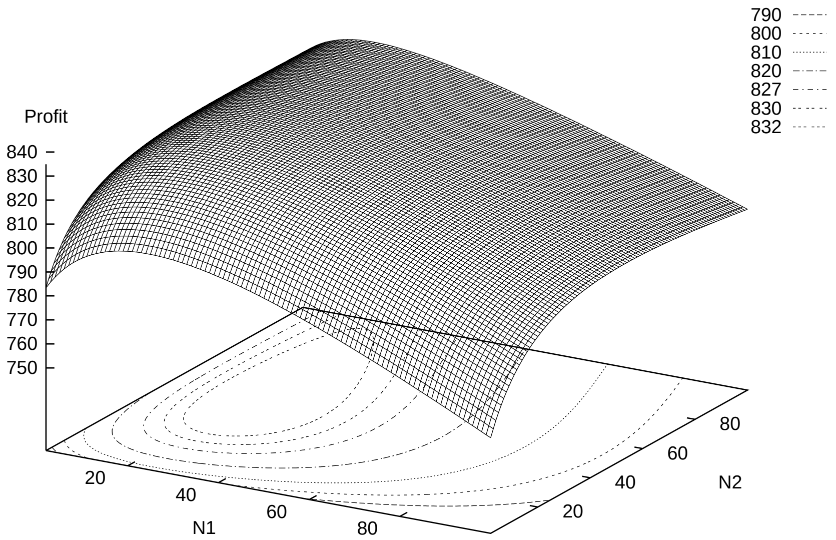

Figure 10. Profit vs. N_1 and N_2

Figure 10 is a graph of $\Pi(N_1, N_2)$ vs. N_1 and N_2 for the balanced three-machine continuous material line with parameters $r_i = 0.1$, $p_i = 0.01$, and $\mu_i = 1$ for $i = 1, 2, 3$. The cost coefficients are $\Phi = 1000$, $a_i = 0$, and $b_i = 1$ for $i = 1, 2$. Therefore $\Pi(N_1, N_2) = 1000P(N_1, N_2) - \bar{n}_1 - \bar{n}_2$. The contour lines indicate that the function $\Pi(N_1, N_2)$ has a global maximum and no other local maxima. The maximum profit for this line is realized when $(N_1, N_2) = (21, 49)$. (These have been rounded to integers.) Note that Figure 10 is not symmetric and the buffer sizes are not equal, even though the line is balanced.

7.4. Profit Maximization Algorithms

The algorithm for the constrained profit maximization problem (10) is essentially the same as the algorithm for the dual problem (Section 4). It is the same as Figure 4 with P replaced by Π.

The algorithm for the unconstrained profit maximization problem (11) follows the primal algorithm (Section 5) very closely. It also consists of the iterated solution of two problems: the constrained profit maximization problem (in place of the dual) and a one-dimensional profit maximization problem (in place of the one-dimensional primal).

Details on the algorithms, as well as numerical examples, appear in Schor (1995).

8. Conclusions

The algorithms described here are fast and accurate. They can be used in the design of production systems. These methods can easily be extended to assembly/disassembly systems (Gershwin, 1994). It would be valuable to extend them to systems that have loops, such as systems with captive pallets or fixtures. Other useful extensions include systems with multiple part types, with set-up changes, etc. The main challenges in such extensions are more in the modeling and analysis than in the optimization algorithm.

These extensions are valuable for designing more general systems. As shown in Gershwin (1997) they are also valuable in designing real-time scheduling policies for manufacturing systems.

References

[1] I. Adan and J. Van Der Wal (1989), "Monotonicity of the Throughput in Single Server Production and Assembly Networks with Respect to the Buffer Sizes," *Queueing Networks with Blocking,* H. G. Perros and T. Altiok, Editors, Elsevier Science Publishers, pp. 345-356, 1989.

[2] T. Altiok and S. Stidham Jr. (1983), "The Allocation of Interstage Buffer Capacities in Production Lines," *IIE Transactions,* **15** pp. 292-299.

[3] V. Anantharam and P. Tsoucas (1990), "Stochastic Concavity of Throughput in Series of Queues with Finite Buffers," *Advances in Applied Probability*, Vol. 22, pp. 761-763, 1990.

[4] D. R. Anderson and C. L. Moodie (1969), "Optimal Buffer Storage Capacity in Production Line Systems," *International Journal of Production Research*, Vol. 7 No. 3, pp. 233-240, 1969.

[5] K. Barten (1962), "A Queueing Simulator for Determining Optimum Inventory Levels in a Sequential Process," *Journal of Industrial Engineering*, Volume 13, No. 4, July-August, 1962, pp. 245-252.

[6] M. Burman (1995), "New Results in Flow Line Analysis", Ph. D. Thesis, MIT, Cambridge MA, 1995.

[7] J. A. Buzacott (1967), "Automatic Transfer Lines with Buffer Stocks," *International Journal of Production Research*, Vol. 5, No. 3, pp. 182-200.

[8] J. A. Buzacott (1968), "Prediction of the Efficiency of Production Systems without Internal Storage," *International Journal of Production Research*, Vol. 6, No. 3, pp. 173-188.

[9] J. A. Buzacott and J. G. Shanthikumar (1993), *Stochastic Models of Manufacturing Systems*, Prentice Hall, 1993.

[10] M. Caramanis (1987), "Production System Design: A Discrete Event Dynamic System and Generalized Benders' Decomposition Approach," *International Journal of Production Research*, Vol. 25, No. 8, pp. 1223-1234, 1987.

[11] We-Min Chow (1987), "Buffer Capacity Analysis for Sequential Production Lines with Variable Processing Times," *International Journal of Production Research*, Vol. 25, No. 8, pp. 1183-1196, 1987.

[12] Y. Dallery, R. David, and X.L. Xie (1988), "An Efficient Algorithm for Analysis of Transfer Lines with Unreliable Machines and Finite Buffers," *IIE Transactions,* Vol. 20, No. 3, pp. 280-283, September, 1988.

[13] Y. Dallery and S. B. Gershwin (1992), "Manufacturing Flow Line Systems: A Review of Models and Analytical Results," *Queueing Systems Theory and Applications, Special Issue on Queueing Models of Manufacturing Systems*, Volume 12, No. 1-2, December, 1992, pp. 3-94.

[14] Y. Dallery, Z. Liu and D. Towsley, "Equivalence, Reversibility, Symmetry and Concavity Properties in Fork/Join Queueing Networks with Blocking, *Journal of the ACM,* Vol. 41, No. 5, pp. 903-942, 1994.

[15] E. Dogan and T. Altiok (1995), "Gradient Estimation of the Output Rate in Transfer Lines," *preprint*, Rutgers University, Dept. of Industrial Engineering.

[16] M. C. Freeman (1964), "The Effects of Breakdowns and Interstage Storage on Production Line Capacity," *Journal of Industrial Engineering*, Volume 15, Number 4, July-August, 1964, pp. 194-200.

[17] S. B. Gershwin (1987), "An Efficient Decomposition Algorithm for The Approximate Evaluation of Tandem Queues with Finite Storage Space and Blocking," *Operations Research*, Vol. 35, pp. 291-305.

[18] S. B. Gershwin (1994), *Manufacturing Systems Engineering,* Prentice-Hall, 1994.

[19] S. B. Gershwin (1997), "Design and Operation of Manufacturing Systems — Control- and System-Theoretical Models and Issue," *1997 American Control Conference*, Albuquerque, New Mexico, June 4 – 6, 1997.

[20] S. B. Gershwin and Y. Goldis (1995), "Efficient Algorithms for Transfer Line Design", MIT Laboratory for Manufacturing and Productivity Report LMP-95-005, 1995.

[21] S. B. Gershwin and I. C. Schick (1980), "Continuous Model of an Unreliable Material Flow Systems with a Finite Interstage Buffer," MIT LIDS Report LIDS-R-1039, September, 1980.

[22] P. Glasserman and D. D. Yao (1996), "Structured Buffer-Allocation Problems," *Journal of Discrete Event Dynamic Systems,* Vol 6, pp. 9-42, 1996.

[23] J. M. Hatcher (1969), "The Effect of Internal Storage on the Production Rate of a Series of Stages Having Exponential Service Times," *AIIE Transactions*, Volume 1, Number 2, June, 1969, pp. 150-156.

[24] F. S. Hillier and K. C. So (1991), "The Effect of Machine Breakdowns and Interstage Storage on the Performance of Production Line Systems," *International Journal of Production*

Research, Vol. 29, No. 10, pp. 2043-2055, 1991.

[25] F. S. Hillier, K. C. So, and R. W. Boling (1993), "Toward Characterizing the Optimal Allocation of Storage Space in Production Line Systems with Variable Processing Times," *Management Science*, Volume 39, Number 1, January, 1993, pp. 126-133.

[26] Y. C. Ho, M. A. Eyler, and T. T. Chien (1979), "A Gradient Technique for General Buffer Storage Design in a Production Line," *International Journal of Production Research*, Vol. 17, No. 6, pp. 557-580.

[27] Y. C. Ho, M. A. Eyler, and T. T. Chien (1983), "A New Approach to Determine Parameter Sensitivities of Transfer Lines," *Management Science*, Volume 29, Number 6, June 1983, pp. 700-714.

[28] D. Jacobs and S. M. Meerkov (1993), "A System-Theoretic Property of Serial Production Lines: Improvability," The University of Michigan College of Engineering Control Group Reports, Report No. CGR-93-1, January, 1993.

[29] D. Jacobs and S. M. Meerkov (1994), "System-Theoretic Properties of the Process of Continuous Improvement in Production Systems," *Proceedings of the American Control Conference*, Baltimore, Maryland, June, 1994, pp. 3323-3327.

[30] M. A. Jafari, J. G. Shanthikumar (1989), "Determination of Optimal Buffer Storage Capacities and Optimal Allocation in Multistage Automatic Transfer Lines," *IIE Transactions*, Vol. 21, No. 2, pp. 130-134, June, 1989.

[31] A. D. Knott (1970), Letter, *AIIE Transactions*, Volume 2, Number 3, September, 1970, p. 273.

[32] S. A. Kraemer and R. F. Love (1970), "A Model for Optimizing the Buffer Inventory Storage Size in a Sequential Production System," *AIIE Transactions*, Vol. II, No. 1, pp. 64-69, March, 1970.

[33] C.-T. Kuo, J.-T. Lim and S. M. Meerkov (1994), "Improvability Analysis of a Machining Transfer Line: An Application," The University of Michigan College of Engineering Control Group Reports, Report No. CGR-94-08, April, 1994.

[34] C.-M. Liu and C.-L. Lin (1994), "Performance Evaluation of Unbalanced Serial Production Lines," *International Journal of Production Research*, Vol. 32, No. 12, pp. 2897-2914.

[35] G. E. Martin (1993), "Predictive Formulae for Unpaced Line Efficiency," *International Journal of Production Research*, Vol. 31, No. 8, pp. 1981-1990, 1993.

[36] G. E. Martin (1994), "Optimal Design of Production Lines," *International Journal of Production Research*, Vol. 32, No. 5, pp. 989-1000, 1994.

[37] L. E. Meester and J. G. Shanthikumar (1990), "Concavity of the Throughput of Tandem Queueing Systems with Finite Buffer Storage Space," *Advances in Applied Probability*, Vol. 22, pp. 764- 767, 1990.

[38] J. A. Muckstadt and S. R. Tayur (1995a), "A Comparison of Alternative Kanban Control Mechanisms. I. Background and Structural Results," *IIE Transactions*, Vol. 27, pp. 140-150, 1995.

[39] J. A. Muckstadt and S. R. Tayur (1995b), "A Comparison of Alternative Kanban Control Mechanisms. II. Experimental Results," *IIE Transactions*, Vol. 27, pp. 151-161, 1995.

[40] T. Park (1993), "A Two-Phase Heuristic Algorithm for Determining Buffer Sizes of Pro-

duction Lines," *International Journal of Production Research*, Vol. 31, No. 3, pp. 613-631, 1993.

[41] E. L. Plambeck, B.-R. Fu, S. M. Robinson, and R. Suri (1993), "Throughput Optimization in Tandem Production Lines Via Nonsmooth Programming," *Proceedings of the 1993 Summer Computer Simulation Conference*, Boston, 1993.

[42] S. G. Powell (1994), "Buffer Allocation in Unbalanced Three-Station Serial Lines," *International Journal of Production Research*, Vol. 32, No. 9, pp. 2201-2218, 1994.

[43] S. G. Powell and D. F. Pyke (1994), "Optimal Allocation of Buffers in Serial Production Lines with a Single Bottleneck," The Amos Tuck School of Business Administration, Dartmouth College, Working Paper No. 301, July, 1994.

[44] J. E. Schor (1995) "Efficient Algorithms for Buffer Allocation", M. S. Thesis, MIT, Cambridge MA, 1995.

[45] D. Seong, S. Y. Chang, and Y. Hong (1994a), "Heuristic Algorithms for Buffer Allocation in a Production Line with Unreliable Machines," Department of Industrial Engineering, POSTECH, Korea; accepted for publication, *International Journal of Production Research.*

[46] D. Seong, S. Y. Chang, and Y. Hong (1994b), "An Algorithm for Buffer Allocation with Linear Resource Constraints in a Continuous Flow Production Line," Department of Industrial Engineering, POSTECH, Korea, Technical Report IE-TR-94-05.

[47] B. A. Sevast'yanov (1962), "Influence of Storage Bin Capacity on the Average Standstill Time of a Production Line," Teoriya Veroyatnostey i ee Primeneniya, Vol. 7, No. 4, pp. 438-447 In Russian). (Eng. tr. Theory of Probability and its Applications, Vol. 7, No. 4 , pp. 429-438 (1962)).

[48] J. G. Shanthikumar and D. D. Yao (1989), "Monotonicity and Concavity Properties in Cyclic Queueing Networks with Finite Buffers," *Queueing Networks with Blocking,* H.G. Perros and T. Altiok, Editors, Elsevier Science Publishers, pp. 325-344, 1989.

[49] T. J. Sheskin (1976) "Allocation of Interstage Storage Along an Automatic Production Line," *AIIE Transactions*, Vol. 8, No. 1, March, 1976, pp 146-152.

[50] J. MacGregor Smith and S. Daskalaki (1988), "Buffer Space Allocation in Automated Assembly Lines," *Operations Research*, Vol. 36, No. 2, pp. 343-357, March-April, 1988.

[51] A. L. Soyster, J. W. Schmidt, and M. W. Rohrer (1979), "Allocation of Buffer Capacities for a Class of Fixed Cycle Production Lines," *AIIE Transactions*, Vol. 11, No. 2, June, 1979, pp. 140-146.

[52] S. R. Tayur (1992), "Properties of Serial Kanban Systems," *Queueing Systems*, Vol. 12, pp. 297-318, 1992

[53] H. Yamashita and T. Altiok (1997), "Buffer Capacity Allocation for a Desired Throughput in Production Lines," Presented at the Workshop on *Performance Evaluation and Optimization of Production Lines,* International Workshop held on May 19-21, 1997, on Samos Island, in the Department of Mathematics, University of the Aegean, GR-832 00 Karlovassi, Samos Island, Greece.