A DYNAMIC PROGRAMMING ALGORITHM FOR THE BUFFER ALLOCATION PROBLEM IN HOMOGENEOUS ASYMPTOTICALLY RELIABLE SERIAL PRODUCTION LINES

A. C. DIAMANTIDIS AND C. T. PAPADOPOULOS

Received 9 February 2004 and in revised form 5 April 2004

In this study, the buffer allocation problem (BAP) in homogeneous, asymptotically reliable serial production lines is considered. A known aggregation method, given by Lim, Meerkov, and Top (1990), for the performance evaluation (i.e., estimation of throughput) of this type of production lines when the buffer allocation is known, is used as an evaluative method in conjunction with a newly developed dynamic programming (DP) algorithm for the BAP. The proposed algorithm is applied to production lines where the number of machines is varying from four up to a hundred machines. The proposed algorithm is fast because it reduces the volume of computations by rejecting allocations that do not lead to maximization of the line's throughput. Numerical results are also given for large production lines.

1. Introduction and literature review

The analysis of production or flow lines and, more generally, of manufacturing systems has been the object of numerous studies. Usually, production lines are modeled as serial queuing networks and analyzed either analytically as Markovian models or via approximate decomposition methods and simulation.

The literature on the modeling of production lines is vast and a large amount of research over the years has been devoted to this area. One of the first most complete analyses of such systems was published in 1962 by Sevastyanov [20]. The large amount of research allows us to review only the most directly relevant studies here.

For a systematic classification of the relevant works on the stochastic modeling of production lines, the interested reader is referred to the books by Buzacott and Shanthikumar [3], Papadopoulos et al. [17], Gershwin [7], Altiok [2], and Helber [10], and the review papers by Dallery and Gershwin [5] and Papadopoulos and Heavey [16], among others.

Hillier and Boling [11] and Heavey et al. [9], analyzed serial production lines using Markovian models, whereas Gershwin [6] and Dallery et al. [4] utilized decomposition approaches to evaluate the performance of the same type of production lines. The former method is practically applicable only to short lines due to the curse of the dimensionality

Mathematical Problems in Engineering 2004:3 (2004) 209–223
2000 Mathematics Subject Classification: 78M50, 90B30, 90C39, 49L20
URL: http://dx.doi.org/10.1155/S1024123X04402014

problem arising in the Markovian models. The latter method can be applied to large production lines at the expense of the accuracy of the numerical results. Altiok [2], among others, studied the phase-type distributions and their use to approximate any general distribution of service or interarrival times in the stochastic modeling of production lines.

Meerkov and his colleagues have developed performability, a quite interesting theory and method of analysis of production lines. Lim et al. [14] presented an asymptotic analysis technique for a model of a serial production line, gave estimates of its accuracy, and formulated a convergence theorem for their solution algorithm.

One of the most interesting questions that designers face in a serial production line is the buffer allocation problem (BAP), that is, how much buffer storage to allow and where to place it within the line. This is an important question because buffers can have a great impact on the efficiency of the production line. Buffers compensate for the blocking and the starving of the line's stations. However, buffer storage is expensive both due to its direct cost and due to the increase of the work-in-process (WIP) inventories. In [17], both evaluative and generative (optimization) models are given for modeling the various types of manufacturing systems.

The BAP has been solved via various techniques such as simulated annealing [21, 22], genetic algorithms [15], cross entropy search method [1], gradient techniques [8], heuristics [18], and other search methods (tabu search, among others).

One of the optimization methods for solving the BAP is the dynamic programming (DP) method. Several authors have employed the DP method for solving the BAP (see Jafari and Shanthikumar [12], Kubat and Sumita [13], and Yamashita and Altiok [23], among others). However, this method was employed in the case of a production line with synchronous transfer as defined in [17], among others, where the steady-state throughput can be approximated in a closed recursive form. Another classification of the research work relevant to the BAP is based on whether the lines under study are balanced or unbalanced. A line is called balanced (or unbalanced) if the mean processing times at each station are equal (or unequal). Powell [19] provided a literature review according to this scheme.

In this paper, a DP algorithm for the BAP is developed. This algorithm makes use of an aggregation procedure to approximate various performance measures of the production line developed by Lim et al. [14].

The remainder of this paper is structured as follows. Section 2 presents the problem and the model. Section 3 presents the proposed DP algorithm with an application example. Section 4 gives numerical results for several production lines, both short and large, and Section 5 concludes the paper and proposes some areas for further research. Finally, the appendix gives the aggregation method for the approximation of throughput for a given buffer allocation, as taken by Lim et al. [14].

2. The model and the problem definition

The main assumptions of our model are the same as those made by Lim et al. [14], as follows.

(1) A serial production line (see Figure 2.1) consists of M machines M_i, $i = 1,\ldots,M$, and $M-1$ buffers B_i of finite capacity at least one (this assumption is dictated by

Figure 2.1. A production line with M machines and $M-1$ buffers.

(A.1), as otherwise function $Q(a,N)$ is not defined). Assuming that each buffer stores at least 1 unit and N is the total buffer capacity that can be allocated, it is obvious that for a production line consisting of M machines and $M-1$ buffers, the maximum buffer capacity that each buffer can store is up to $N-M+2$ units.

(2) An inexhaustive supply of workpieces is available upstream machine M_1, and an unlimited storage area is present downstream machine M_M. Thus the first machine is never starved and the last machine is never blocked.

(3) All machines have equal and constant service times. Time is scaled so that this machine cycle takes one time unit. Thus processing times are assumed to be deterministic and identical for all machines and are taken as the time unit.

(4) Machine M_i, being not blocked and not starved, takes part in production during any time slot with probability δ_i and fails to do so with probability $1-\delta_i$ (production lines in which machines have this property are called homogeneous by Lim et al. [14], that is, a homogeneous line is characterized by machines with one parameter only, δ_i, instead of the two parameters p_i and r_i usually used in the literature, where p_i and r_i denote, respectively, the failure rate and the repair rate of machine M_i). It is assumed that blocked and/or starved machines do not fail.

(5) It is assumed that the production lines under consideration are homogeneous, asymptotically reliable, that is, $\delta_i = 1-\varepsilon\Lambda_i$, where $0<\varepsilon\ll 1$ and Λ_i, $i=1,\ldots,M$, is independent of ε. The Λ_i's are known as the *loss parameters*, as defined by Lim et al. [14].

Following the lines of Lim et al. [14], denote by L_i the cumulative losses in the ith operation (jobs/h) and by AX_i the actual throughput, then $\delta_i = 1-L_i/AX_i$, and since $\delta_i = 1-\varepsilon\Lambda_i$, where $0<\varepsilon\ll 1$, it is obvious that $L_i \ll AX_i$. Thus the loss parameter Λ_i refers to the fraction of the cumulative losses in the ith operation divided by the actual throughput of machine M_i.

Major decisions in designing production lines involve the workload allocation and the BAPs with respect to an objective function such as profit maximization or throughput maximization for a given total buffer capacity. In this study, the latter has been chosen to deal with; namely, our objective is to find the optimal buffer allocation for a given total buffer capacity in order to maximize the average production rate of the production line. The above problem may be expressed mathematically as follows.

Find the optimal vector $B=(N_1,\ldots,N_{M-1})$ that maximizes $\{X_M\}$ given that $\sum_{j=1}^{M-1} N_j = N$, where X_M is the mean production rate (throughput) of a production line consisting of M machines and $M-1$ buffers. N is the total buffer capacity and N_i is the capacity of buffer i, $i=1,\ldots,M-1$.

3. The dynamic programming algorithm

Before expressing the DP algorithm mathematically, we introduce the following symbols.

Z_i is the buffer capacity that the DP algorithm allocates to buffer i and to all buffers upstream buffer i. Therefore $Z_i = N_1 + \cdots + Ni$, for all $i \geq 2$. It is obvious that $Z_1 = N_1$ and that $Z_{M-1} = N_1 + \cdots + N_{M-1} = N$.

$\Lambda_j^f(N_{j-1})$ is the value of the aggregated loss parameter Λ_j^f, defined in the appendix, when buffer space N_{j-1} is allocated to buffer $j-1$, where $N_{j-1} = 1,2,\ldots,N-M+2$, $j = 2,\ldots,M$. Equation (A.14) in the appendix indicates that parameter $\Lambda_{j-1}^f(N_{j-1})$ is used in order to calculate parameter Λ_j^f. From the various values of parameter Λ_{j-1}^f, that one with the maximum value is selected.

$f_1(Z_1)$ is the set of feasible values of parameter $\Lambda_2^f(N_1)$, $N_1 = 1,\ldots,N-M+2$, that are used to calculate $f_2(Z_2)$.

The DP algorithm consists of four steps which are summarized below.

Step 1. Calculate the forward pass loss parameters $\Lambda_i^f(N_{i-1})$, $i = 2,\ldots,M$, $N_{i-1} = 1,\ldots,N-M+2$, by using expression (A.9) of Lim et al. [14].

Step 2. The fundamental recursion equation. Execute the following recursive equations:

$$
\begin{aligned}
f_j(Z_j) &= \min_{\substack{\text{feasible values} \\ N_j, f_{j-1}(Z_{j-1})}} \left[\Lambda_{j+1}^f(N_j) + f_{j-1}(Z_{j-1})\right], \\
j &= 2,\ldots,M-1, \quad N_j = 1,\ldots,N-M+2, \\
Z_j &= j,\ldots,N-M+j+1, \qquad Z_{M-1} = N, \\
Z_{j-1} &= Z_j - N_j.
\end{aligned}
\tag{3.1}
$$

Step 3. Termination of the algorithm. The algorithm terminates when the value of $f_{M-1}(Z_{M-1})$ is calculated.

Step 4. Determination of the optimal buffer allocation. The optimal buffer allocation is given by vector $(T_1, T_2, \ldots, T_{M-1})$, with its elements obtained as follows.

(1) Allocate T_{M-1} units of buffer space to buffer B_{M-1}, where T_{M-1} is the value for which $f_{M-1}(N)$ is obtained.
(2) Allocate T_{M-2} units of buffer space to buffer B_{M-2}, where T_{M-2} is the value for which $f_{M-2}(N - T_{M-1})$ is obtained.
(3) Allocate T_{M-3} units of buffer space to buffer B_{M-3}, where T_{M-3} is the value for which $f_{M-3}(N - T_{M-1} - T_{M-2})$ is obtained, and so forth.
(4) Allocate T_2 and T_1 units of buffer capacity to buffers B_2 and B_1, respectively, where T_2 and T_1 are the values for which $f_2(N - (T_{M-1} + T_{M-2} + \cdots + T_3))$ is obtained.

3.1. Application of the algorithm: an example. In this section, an application of the algorithm is given for a production line with 4 machines and *loss parameters* $\Lambda_1 = 3.4$, $\Lambda_2 = 2.1$, $\Lambda_3 = 4.3$, and $\Lambda_4 = 1.1$, respectively. The total buffer capacity that is to be allocated is 10 units.

Table 3.1. Terms $\Lambda_j^f(N_{j-1})$, $j = 2,3,4$, calculated as described in Step 1.

Buffer 1		Buffer 2		Buffer 3	
N_1	$\Lambda_2^f(N_1)$	N_2	$\Lambda_3^f(N_2)$	N_3	$\Lambda_4^f(N_3)$
1	5.5	1	9.8	1	10.9
2	4.2021	2	7.3874	2	9.9114
3	3.8008	3	6.5986	3	9.8135
4	3.6215	4	6.2158	4	9.8021
5	3.5285	5	5.9952	5	9.8013
6	3.4765	6	5.8553	6	9.80045
7	3.4463	7	5.76104	7	9.800442
8	3.4283	8	5.6948	8	9.800440

Step 1. In this step, we calculate the forward pass loss parameters $\Lambda_i^f(N_{i-1})$, $i = 2,\ldots,4$, $N_{i-1} = 1,\ldots,8$. These parameters are presented in Table 3.1 and will be used throughout the algorithm.

Step 2. In DP, computations are carried out in stages by breaking down the problem into subproblems. Each subproblem is then considered separately with the objective of reducing the volume of computations. However, since the subproblems are interdependent, a procedure must be devised to link the computations in a manner that guarantees that a feasible solution for each stage is also feasible for the entire problem.

$$j = 2, \qquad Z_2 = 2,\ldots,9. \tag{3.2}$$

For $Z_2 = 2 = N_1 + N_2$, the only feasible values of N_1 and N_2 are $\{N_1 = 1 \text{ and } N_2 = 1\}$. As $f_1(Z_1)$, the value of parameter $\Lambda_2^f(1)$ is used and

$$f_2(Z_2 = 2) = \min\{\Lambda_3^f(1) + \Lambda_2^f(1)\} = \min\{9.8 + 5.5\} = 15.3 \tag{3.3}$$

corresponding to $T_1 = 1$ and $T_2 = 1$.

For $Z_2 = 3 = N_1 + N_2$, the feasible values of N_1 and N_2 are $\{N_1 = 1 \text{ and } N_2 = 2\}$ or $\{N_1 = 2 \text{ and } N_2 = 1\}$. As $f_1(Z_1)$, the values of parameters $\Lambda_2^f(1)$, $\Lambda_2^f(2)$ are used and

$$\begin{aligned} f_2(Z_2 = 3) &= \min\{\Lambda_3^f(1) + \Lambda_2^f(2), \Lambda_3^f(2) + \Lambda_2^f(1)\} \\ &= \min\{9.8 + 4.2021, 7.3874 + 5.5\} \\ &= \min\{14.0021, 12.8874\} = 12.8874 \end{aligned} \tag{3.4}$$

corresponding to $T_1 = N_1 = 1$ and $T_2 = N_2 = 2$.

For $Z_2 = 4 = N_1 + N_2$, the feasible values of N_1 and N_2 are $\{N_1 = 1 \text{ and } N_2 = 3\}$, $\{N_1 = 2 \text{ and } N_2 = 2\}$, or $\{N_1 = 3 \text{ and } N_2 = 1\}$. As $f_1(Z_1)$, the values of parameters $\Lambda_2^f(1)$, $\Lambda_2^f(2)$, $\Lambda_2^f(3)$ are used and

$$\begin{aligned} f_2(Z_2 = 4) &= \min\{\Lambda_3^f(1) + \Lambda_2^f(3), \Lambda_3^f(2) + \Lambda_2^f(2), \Lambda_3^f(3) + \Lambda_2^f(1)\} \\ &= \min\{9.8 + 3.8008, 7.3874 + 4.2021, 6.5986 + 5.5\} \\ &= \min\{13.6008, 11.5895, 12.0986, \} = 11.5895 \end{aligned} \tag{3.5}$$

corresponding to $T_1 = N_1 = 2$ and $T_2 = N_2 = 2$.

For $Z_2 = 5 = N_1 + N_2$, the feasible values of N_1 and N_2 are $\{N_1 = 1 \text{ and } N_2 = 4\}$, $\{N_1 = 2 \text{ and } N_2 = 3\}$, $\{N_1 = 3 \text{ and } N_2 = 2\}$, or $\{N_1 = 4 \text{ and } N_2 = 1\}$. As $f_1(Z_1)$, the values of parameters $\Lambda_2^f(1)$, $\Lambda_2^f(2)$, $\Lambda_2^f(3)$, $\Lambda_2^f(4)$ are used and

$$\begin{aligned} f_2(Z_2 = 5) &= \min\{\Lambda_3^f(1) + \Lambda_2^f(4), \Lambda_3^f(2) + \Lambda_2^f(3), \Lambda_3^f(3) + \Lambda_2^f(2), \Lambda_3^f(4) + \Lambda_2^f(1)\} \\ &= \min\{9.8 + 3.6215, 7.3874 + 3.8008, 6.5986 + 4.2021, 6.2158 + 5.5\} \\ &= \min\{13.4215, 11.1882, 10.8007, 11.7158\} = 10.8007 \end{aligned} \tag{3.6}$$

corresponding to $T_1 = N_1 = 2$ and $T_2 = N_2 = 3$.

For $Z_2 = 6 = N_1 + N_2$, the feasible values of N_1 and N_2 are $\{N_1 = 1 \text{ and } N_2 = 5\}$, $\{N_1 = 2 \text{ and } N_2 = 4\}$, $\{N_1 = 3 \text{ and } N_2 = 3\}$, $\{N_1 = 4 \text{ and } N_2 = 2\}$, or $\{N_1 = 5 \text{ and } N_2 = 1\}$. As $f_1(Z_1)$, the values of parameters $\Lambda_2^f(1)$, $\Lambda_2^f(2)$, $\Lambda_2^f(3)$, $\Lambda_2^f(4)$, $\Lambda_2^f(5)$ are used and

$$\begin{aligned} f_2(Z_2 = 6) &= \min\{\Lambda_3^f(1) + \Lambda_2^f(5), \Lambda_3^f(2) + \Lambda_2^f(4), \Lambda_3^f(3) + \Lambda_2^f(3), \\ &\qquad \Lambda_3^f(4) + \Lambda_2^f(2), \Lambda_3^f(5) + \Lambda_2^f(1)\} \\ &= \min\{9.8 + 3.5285, 7.3874 + 3.6215, 3.8008 + 6.5986, \\ &\qquad 6.2158 + 4.2021, 5.9952 + 5.5\} \\ &= \min\{13.3285, 11.0089, 10.3994, 10.4179, 11.4952\} \\ &= 10.3994 \end{aligned} \tag{3.7}$$

corresponding to $T_1 = N_1 = 3$ and $T_2 = N_2 = 3$.

For $Z2 = 7 = N_1 + N_2$, the feasible values of N_1 and N_2 are $\{N_1 = 1 \text{ and } N_2 = 6\}$, $\{N_1 = 2 \text{ and } N_2 = 5\}$, $\{N_1 = 3 \text{ and } N_2 = 4\}$, $\{N_1 = 4 \text{ and } N_2 = 3\}$, $\{N_1 = 5 \text{ and } N_2 = 2\}$, or $\{N_1 = 6 \text{ and } N_2 = 1\}$. As $f_1(Z_1)$, the values of parameters $\Lambda_2^f(1)$, $\Lambda_2^f(2)$, $\Lambda_2^f(3)$, $\Lambda_2^f(4)$, $\Lambda_2^f(5)$, $\Lambda_2^f(6)$ are used and

$$\begin{aligned} f_2(Z_2 = 7) &= \min\{\Lambda_3^f(1) + \Lambda_2^f(6), \Lambda_3^f(2) + \Lambda_2^f(5), \Lambda_3^f(3) + \Lambda_2^f(4), \\ &\qquad \Lambda_3^f(4) + \Lambda_2^f(3), \Lambda_3^f(5) + \Lambda_2^f(2), \Lambda_3^f(6) + \Lambda_2^f(1)\} \\ &= \min\{9.8 + 3.4765, 7.3874 + 3.5285, 6.5986 + 3.6215, \\ &\qquad 6.2158 + 3.8008, 5.9952 + 4.2021, 5.8553 + 5.5\} \\ &= \min\{13.2765, 10.9159, 10.2201, 10.0166, 10.1973, 11.3553\} \\ &= 10.0166 \end{aligned} \tag{3.8}$$

corresponding to $T_1 = N_1 = 3$ and $T_2 = N_2 = 4$.

For $Z_2 = 8 = N_1 + N_2$, the feasible values of N_1 and N_2 are $\{N_1 = 1 \text{ and } N_2 = 7\}$, $\{N_1 = 2 \text{ and } N_2 = 6\}$, $\{N_1 = 3 \text{ and } N_2 = 5\}$, $\{N_1 = 4 \text{ and } N_2 = 4\}$, $\{N_1 = 5 \text{ and } N_2 = 3\}$, $\{N_1 = 6 \text{ and } N_2 = 2\}$, or $\{N_1 = 7 \text{ and } N_2 = 1\}$. As $f_1(Z_1)$, the values of parameters $\Lambda_2^f(1)$, $\Lambda_2^f(2)$, $\Lambda_2^f(3)$, $\Lambda_2^f(4)$, $\Lambda_2^f(5)$, $\Lambda_2^f(6)$, $\Lambda_2^f(7)$ are used and

$$\begin{aligned} f_2(Z_2 = 8) &= \min\{\Lambda_3^f(1) + \Lambda_2^f(7), \Lambda_3^f(2) + \Lambda_2^f(6), \Lambda_3^f(3) + \Lambda_2^f(5), \Lambda_3^f(4) + \Lambda_2^f(4), \\ &\qquad \Lambda_3^f(5) + \Lambda_2^f(3), \Lambda_3^f(6) + \Lambda_2^f(2), \Lambda_3^f(7) + \Lambda_2^f(1)\} \\ &= \min\{9.8 + 3.4463, 7.3874 + 3.4765, 3.65986 + 5.285, 6.2158 + 3.6215, \\ &\qquad 5.9952 + 3.8008, 5.8553 + 4.2021, 5.76104 + 5.5\} \\ &= \min\{13.2463, 10.8639, 10.1271, 9.8373, 9.796, 10.0574, 11.26104\} \\ &= 9.796 \end{aligned} \tag{3.9}$$

corresponding to $T_1 = N_1 = 3$ and $T_2 = N_2 = 5$.

For $Z_2 = 9 = N_1 + N_2$, the feasible values for N_1 and N_2 are $\{N_1 = 1 \text{ and } N_2 = 8\}$, $\{N_1 = 2 \text{ and } N_2 = 7\}$, $\{N_1 = 3 \text{ and } N_2 = 6\}$, $\{N_1 = 4 \text{ and } N_2 = 5\}$ $\{N_1 = 5 \text{ and } N_2 = 4\}$, $\{N_1 = 6 \text{ and } N_2 = 3\}$, $\{N_1 = 7 \text{ and } N_2 = 2\}$, or $\{N_1 = 8 \text{ and } N_2 = 1\}$. As $f_1(Z_1)$, the values of parameters $\Lambda_2^f(1)$, $\Lambda_2^f(2)$, $\Lambda_2^f(3)$, $\Lambda_2^f(4)$, $\Lambda_2^f(5)$, $\Lambda_2^f(6)$, $\Lambda_2^f(7)$, $\Lambda_2^f(8)$ are used.

Therefore

$$\begin{aligned} f_2(Z_2 = 9) &= \min\{\Lambda_3^f(1) + \Lambda_2^f(8), \Lambda_3^f(2) + \Lambda_2^f(7), \Lambda_3^f(3) + \Lambda_2^f(6), \Lambda_3^f(4) + \Lambda_2^f(5), \\ &\qquad \Lambda_3^f(5) + \Lambda_2^f(4), \Lambda_3^f(6) + \Lambda_2^f(3), \Lambda_3^f(7) + \Lambda_2^f(2), \Lambda_3^f(8) + \Lambda_2^f(1)\} \\ &= \min\{9.8 + 3.4283, 7.3874 + 3.4463, 6.5986 + 3.4765, 6.2158 + 3.5285, \\ &\qquad 5.9952 + 3.6215, 5.8553 + 3.8008, 5.76104 + 4.2021, 5.6948 + 5.5\} \\ &= \min\{13.2283, 10.8337, 10.0751, 9.7443, 9.6167, 9.6561, 9.96314, 11.1948\} \\ &= 9.6167 \end{aligned} \tag{3.10}$$

corresponding to $T_1 = N_1 = 4$ and $T_2 = N_2 = 5$;

$$i = 3, \qquad Z_3 = 10. \tag{3.11}$$

For $N_3 = 1, \ldots, N - M + 2 = 1, \ldots, 8$, it is straightforward that the feasible values of $f_2(Z_2)$ and N_3 are $(N_3 = 1 \text{ and } f_2(Z_2 = 9))$, $(N_3 = 2 \text{ and } f_2(Z_2 = 8))$, $(N_3 = 3 \text{ and } f_2(Z_2 = 7))$, $(N_3 = 4 \text{ and } f_2(Z_2 = 6))$, $(N_3 = 5 \text{ and } f_2(Z_2 = 5))$, $(N_3 = 6 \text{ and } f_2(Z_2 = 4))$, $(N_3 = 7 \text{ and } f_2(Z_2 = 3))$, or $(N_3 = 8 \text{ and } f_2(Z_2 = 2))$.

Therefore

$$\begin{aligned} f_3(Z_3 = 10) = \min\{&\Lambda_4^f(1) + f_2(Z_2 = 9), \Lambda_4^f(2) + f_2(Z_2 = 8), \Lambda_4^f(3) + f_2(Z_2 = 7), \\ &\Lambda_4^f(4) + f_2(Z_2 = 6), \Lambda_4^f(5) + f_2(Z_2 = 5), \Lambda_4^f(6) + f_2(Z_2 = 4), \\ &\Lambda_4^f(7) + f_2(Z_2 = 3), \Lambda_4^f(8) + f_2(Z_2 = 2)\} \end{aligned}$$

$$
\begin{aligned}
&= \min\{10.9 + 9.6167, 9.9114 + 9.796, 10.0166 + 9.8135,\\
&\qquad 10.3994 + 9.8021, 10.8007 + 9.8013, 11.5895 + 9.80045,\\
&\qquad 12.8874 + 9.800442, 15.3 + 9.800440\}\\
&= 19.7074
\end{aligned}
\tag{3.12}
$$

corresponding to $T_3 = N_3 = 2$ and $f_2(Z_2 = 8)$.
Step 3. The algorithm terminates as $f_3(Z_3)$ has been calculated.
Step 4. From the previous calculations, notice that $f_2(Z_2 = N - T_3 = 10 - 2 = 8)$ was obtained for $T_1 = 3$ and $T_2 = 5$.

Therefore the optimal buffer allocation is given by the vector $(T_1, T_2, T_3) = (3,5,2)$.

4. Numerical results

In this section, numerical results are presented showing buffer allocations obtained using the proposed DP algorithm for four, five, six stations, and large production lines with up to a hundred stations (the latter are given in Section 4.1). For the short lines, by the enumeration method, all possible buffer allocations of a given total buffer capacity were tested and the optimal buffer allocation, namely, that one giving the maximum throughput, was obtained. The DP algorithm was implemented in PASCAL and in a very slow old PC486 system. Tables 4.1, 4.2, and 4.3 present the buffer allocations, for given total buffer capacities, obtained by the DP algorithm for short lines with four, five, and six stations, respectively.

Comment. We have applied enumeration for all total buffer capacities given in Table 4.1, that is, for 4 to 10 units of total buffer capacities. Comparing the results from the enumeration method with those obtained by the DP algorithm, we have found that in all cases the results were identical. Also notice that the run time is very small and lies between 0.04 and 0.19 seconds even in a slow old PC486 system. In Tables 4.2 and 4.3, for the cases where the results from the enumeration method differ from those obtained by the proposed algorithm, a percentage error has been introduced. The error has been calculated using the following formula:

$$
\text{Error} = \frac{\left| X_{M,\text{enum}} - X_{M,\text{DP}} \right|}{X_{M,\text{enum}}} \times 100\%, \tag{4.1}
$$

where $X_{M,\text{enum}}$ and $X_{M,\text{DP}}$ denote the throughput of the buffer configuration obtained by enumeration and the proposed DP algorithm, respectively.

4.1. Numerical results for large production lines. In this section, numerical results are presented, showing buffer allocations obtained using the proposed DP algorithm in production lines with many stations M, $10 \le M \le 100$. Tables 4.4, 4.5, 4.6, and 4.7 present the buffer allocations obtained by the DP algorithm for given total buffer capacities, for large production lines with ten, fifty, eighty, and one hundred stations, respectively.

Table 4.1. Application of the DP algorithm in a four-station production line with $\Lambda_1 = 3.4$, $\Lambda_2 = 2.1$, $\Lambda_3 = 4.3$, and $\Lambda_4 = 1.1$.

N	Buffer 1	Buffer 2	Buffer 3	X_4 ($\varepsilon = 0.01$)	Run time
10	3	5	2	0.9522	0.19 s
9	3	4	2	0.9510	0.17 s
8	3	3	2	0.9487	0.11 s
7	2	3	2	0.9452	0.09 s
6	2	2	2	0.9396	0.08 s
5	2	2	1	0.9328	0.06 s
4	1	2	1	0.9182	0.04 s

Table 4.2. Application of the DP algorithm in a five-station production line with $\Lambda_1 = 3.4$, $\Lambda_2 = 2.1$, $\Lambda_3 = 4.3$, $\Lambda_4 = 1.1$, and $\Lambda_5 = 5.5$.

N	$B1$	$B2$	$B3$	$B4$	$X_{5,\text{enum}}$	$X_{5,\text{DP}}$	Error	Run time
10	2	3	2	3	0.9359	0.9358	0.0106%	0.09 s
9	2	2	2	3	0.9321	0.9320	0.0107%	0.08 s
8	2	2	1	3	0.9269	0.9201	0.7278%	0.06 s
7	2	2	1	2	0.9164	0.9114	0.5370%	0.05 s
6	1	2	1	2	0.9009	0.9001	0.0872%	0.05 s

Table 4.3. Application of the DP algorithm in a six-station production line with $\Lambda_1 = 3.4$, $\Lambda_2 = 2.1$, $\Lambda_3 = 4.3$, $\Lambda_4 = 1.1$, $\Lambda_5 = 2.4$, and $\Lambda_6 = 3.7$.

N	$B1$	$B2$	$B3$	$B4$	$B5$	$X_{6,\text{enum}}$	$X_{6,\text{DP}}$	Error	Run time
10	2	2	2	2	2	0.9338	0.9338	0%	1.89 s
9	2	2	1	2	2	0.9237	0.9217	0.212%	0.91 s
8	1	2	1	2	2	0.9139	0.9096	0.471%	0.73 s
7	1	2	1	1	2	0.8969	0.8907	0.686%	0.49 s
6	1	1	1	1	2	0.8715	0.8589	1.441%	0.30 s

Table 4.4. Application of the DP algorithm in a ten-station production line with $M = 10$, $N = 20$. The production rate for this specific buffer allocation is 0.9382.

Λ_1	Λ_2	Λ_3	Λ_4	Λ_5	Λ_6	Λ_7	Λ_8	Λ_9	Λ_{10}
3.4	2.1	4.3	1.1	2.4	3.7	1.5	2.3	2.6	1.4

$B1$	$B2$	$B3$	$B4$	$B5$	$B6$	$B7$	$B8$	$B9$	Time
2	3	2	2	3	2	2	2	2	0.3 s

Table 4.5. Application of the DP algorithm in a fifty-station production line with $M = 50$, $N = 90$. The production rate for this specific buffer allocation is 0.8651.

$B1$	$B2$	$B3$	$B4$	$B5$	$B6$	$B7$	$B8$	$B9$	$B10$
2	2	1	2	2	1	2	2	1	2
$B11$	$B12$	$B13$	$B14$	$B15$	$B16$	$B17$	$B18$	$B19$	$B20$
2	2	2	2	2	2	1	2	2	2
$B21$	$B22$	$B23$	$B24$	$B25$	$B26$	$B27$	$B28$	$B29$	$B30$
2	2	2	2	2	2	2	2	2	2
$B31$	$B32$	$B33$	$B34$	$B35$	$B36$	$B37$	$B38$	$B39$	$B40$
2	2	2	2	2	2	2	1	2	2
$B41$	$B42$	$B43$	$B44$	$B45$	$B46$	$B47$	$B48$	$B49$	Time
2	2	2	2	2	2	1	2	1	27 s

Table 4.6. Application of the algorithm in an eighty-station production line with $M = 80$, $N = 200$. The run time is 6 minutes and 0.44 second in a PC486. The production rate for this specific buffer allocation is 0.8810.

$B1$	$B2$	$B3$	$B4$	$B5$	$B6$	$B7$	$B8$	$B9$	$B10$
6	8	2	3	4	3	3	3	2	3
$B11$	$B12$	$B13$	$B14$	$B15$	$B16$	$B17$	$B18$	$B19$	$B20$
3	3	2	2	3	2	2	2	2	3
$B21$	$B22$	$B23$	$B24$	$B25$	$B26$	$B27$	$B28$	$B29$	$B30$
3	3	3	3	3	3	2	3	3	3
$B31$	$B32$	$B33$	$B34$	$B35$	$B36$	$B37$	$B38$	$B39$	$B40$
3	2	2	3	3	3	3	2	2	2
$B41$	$B42$	$B43$	$B44$	$B45$	$B46$	$B47$	$B48$	$B49$	$B50$
2	2	2	2	2	2	2	2	2	2
$B51$	$B52$	$B53$	$B54$	$B55$	$B56$	$B57$	$B58$	$B59$	$B60$
2	2	3	3	3	2	2	2	2	2
$B61$	$B62$	$B63$	$B64$	$B65$	$B66$	$B67$	$B68$	$B69$	$B70$
2	3	2	2	2	2	3	3	3	2
$B71$	$B72$	$B73$	$B74$	$B75$	$B76$	$B77$	$B78$	$B79$	
2	3	2	3	3	2	2	2	3	

Unfortunately, we cannot compare these results with those obtained from enumeration because it is impossible to use enumeration in large production lines (because of the huge number of states that should be examined).

Table 4.7. Application of the DP algorithm in a hundred-station production line with $M = 100$, $N = 400$. The run time is 24 minutes and 6.81 seconds in a PC486. The production rate for this specific buffer allocation is 0.9112.

$B1$	$B2$	$B3$	$B4$	$B5$	$B6$	$B7$	$B8$	$B9$	$B10$
9	14	5	7	8	5	5	5	4	6
$B11$	$B12$	$B13$	$B14$	$B15$	$B16$	$B17$	$B18$	$B19$	$B20$
6	6	4	4	4	4	3	4	4	4
$B21$	$B22$	$B23$	$B24$	$B25$	$B26$	$B27$	$B28$	$B29$	$B30$
5	5	5	5	5	4	4	4	4	4
$B31$	$B32$	$B33$	$B34$	$B35$	$B36$	$B37$	$B38$	$B39$	$B40$
4	4	4	5	5	4	5	3	4	4
$B41$	$B42$	$B43$	$B44$	$B45$	$B46$	$B47$	$B48$	$B49$	$B50$
4	4	3	3	3	4	3	4	3	4
$B51$	$B52$	$B53$	$B54$	$B55$	$B56$	$B57$	$B58$	$B59$	$B60$
3	3	4	4	4	3	3	3	4	4
$B61$	$B62$	$B63$	$B64$	$B65$	$B66$	$B67$	$B68$	$B69$	$B70$
3	4	4	3	4	4	4	4	4	4
$B71$	$B72$	$B73$	$B74$	$B75$	$B76$	$B77$	$B78$	$B79$	$B80$
4	4	3	4	4	4	3	3	4	4
$B81$	$B82$	$B83$	$B84$	$B85$	$B86$	$B87$	$B88$	$B89$	$B90$
4	3	3	3	3	3	3	3	4	3
$B91$	$B92$	$B93$	$B94$	$B95$	$B96$	$B97$	$B98$	$B99$	
3	4	3	4	3	3	3	3	3	

5. Conclusions and further research

In this study, we present a dynamic programming algorithm that solves the buffer allocation problem (BAP) of N units of total buffer capacity in a homogeneous asymptotically reliable serial production line consisting of M machines and $M - 1$ buffers. The main conclusions are as follows.

(1) The proposed dynamic programming algorithm for short (with $M < 10$ stations) production lines found, in almost all cases, the optimal solution for the BAP. In the cases where the algorithm did not give the optimal solution, it gave a near-optimal solution.

(2) The algorithm is quite fast and in all cases where we applied it, we did not encounter any bugs and the algorithm always converged to a solution. The run time in all cases was quite small.

(3) The DP algorithm can be applied in large production lines to effectively (rapidly and accurately) find a near-optimal solution to the BAP. Even in large systems, the proposed algorithm worked quite effectively.

A further investigation to improve the accuracy of the proposed algorithm might include the effect the backward pass search might have on the accuracy of the numerical results. However, an application of this backward pass to a few short lines showed no further improvement in the optimal buffer allocation. Another point that needs further investigation in order to improve the accuracy of the proposed algorithm might be the appropriate use of the values of the loss parameters, Λ_i. In our numerical results, we used values analogous to those used by Lim et al. [14] for short lines.

An extension of the work presented in this paper would be the study of a production line where the general topology would be quite different from the topology of the model considered in this study. That is, a production line which consists of identical machines in parallel order at each workstation. Another optimization problem, apart from the buffer allocation, would be the server allocation as well as the workload allocation problem or, even better, the simultaneous optimization of the parameters of these design problems taken in various combinations.

Appendix

Throughput approximation of homogeneous asymptotically reliable production lines for a given buffer allocation (taken from Lim et al. [14])

The method for the performance evaluation of a homogeneous asymptotically reliable serial production line presented here is introduced and explicitly analyzed in [14].

Define the function

$$Q(a,N) = \frac{1-a}{1-a^N}, \quad a \in \mathbb{R}^+,\ N \in [1,\infty]. \tag{A.1}$$

Two-machine lines. A two-machine, one-buffer production line in steady state is equivalent to a single aggregated machine characterized by

$$\delta_{\text{aggregation}} = 1 - \left[\Lambda_2 + \Lambda_1 Q\left(\frac{\Lambda_2}{\Lambda_1}, N\right)\right]\varepsilon. \tag{A.2}$$

Thus, the loss parameter of the equivalent aggregated machine is

$$\Lambda_{\text{aggregation}} = \Lambda_2 + \Lambda_1 Q\left(\frac{\Lambda_2}{\Lambda_1}, N\right), \tag{A.3}$$

where Λ_1 and Λ_2 are the loss parameters of the first and second machines, respectively. It has been proved that the mean production rate X_2 of a two-machine, one-buffer production line is given by

$$\begin{aligned} X_2 &= 1 - \left[\Lambda_2 + \Lambda_1 Q\left(\frac{\Lambda_2}{\Lambda_1}, N\right)\right]\varepsilon + O(\varepsilon^2) \\ &= 1 - \left[\Lambda_1 + \Lambda_2 Q\left(\frac{\Lambda_1}{\Lambda_2}, N\right)\right]\varepsilon + O(\varepsilon^2). \end{aligned} \tag{A.4}$$

It is straightforward that

$$\Lambda_{\text{aggregation}} = \Lambda_1 + \Lambda_2 Q\left(\frac{\Lambda_1}{\Lambda_2}, N\right). \tag{A.5}$$

Equations (A.3) and (A.5) show that

$$\Lambda_{\text{aggregation}} = \Lambda_1 + \Lambda_2 Q\left(\frac{\Lambda_1}{\Lambda_2}, N\right) = \Lambda_2 + \Lambda_1 Q\left(\frac{\Lambda_2}{\Lambda_1}, N\right). \tag{A.6}$$

The above process can be generalized for the case of a homogeneous asymptotically reliable serial production line consisting of M machines and $M-1$ buffers. Firstly, the first two machines M_1 and M_2 are combined into an aggregated machine with the loss parameter ${\Lambda_2}^f$ defined by (A.3), that is,

$$\Lambda_2^f = \Lambda_2 + \Lambda_1 Q\left(\frac{\Lambda_2}{\Lambda_1}, N_1\right). \tag{A.7}$$

The superscript "f" indicates that during the aggregation, we move forward (from machine M_1 to machine M_M). The aggregated machine, characterized by ${\Lambda_2}^f$, is now combined with the third machine defined by the loss parameter Λ_3. The new aggregated machine is characterized by the loss parameter

$$\Lambda_3^f = \Lambda_3 + \Lambda_2^f Q\left(\frac{\Lambda_3}{\Lambda_2^f}, N_2\right). \tag{A.8}$$

At the ith step of this multistage aggregation process, one may obtain

$$\Lambda_i^f = \Lambda_i + \Lambda_{i-1}^f Q\left(\frac{\Lambda_i}{\Lambda_{i-1}^f}, N_{i-1}\right), \tag{A.9}$$

and at the final step,

$$\Lambda_M^f = \Lambda_M + \Lambda_{M-1}^f Q\left(\frac{\Lambda_M}{\Lambda_{M-1}^f}, N_{M-1}\right). \tag{A.10}$$

The estimate of the throughput obtained as a result of this aggregation is

$$X_M{}^f = 1 - \left[\Lambda_M + \Lambda_{M-1}^f Q\left(\frac{\Lambda_M}{\Lambda_{M-1}^f}, N_{M-1}\right)\right]\varepsilon. \tag{A.11}$$

Because there is no proof that $X_M{}^f$ is close to the real throughput of a production line with M machines and $M-1$ buffers, another set of iterations, this time directed backward instead of forward, should be supplemented. This scheme is called backward aggregation

and aggregates the line moving from the last machine M_M to the first machine M_1. Thus

$$\begin{aligned}
\Lambda^b_{M-1} &= \Lambda_{M-1} + \Lambda_M Q\left(\frac{\Lambda^f_{M-1}}{\Lambda_M}, N_{M-1}\right),\\
\Lambda^b_{M-2} &= \Lambda_{M-2} + \Lambda^b_{M-1} Q\left(\frac{\Lambda^f_{M-2}}{\Lambda^b_{M-1}}, N_{M-2}\right),\\
&\vdots\\
\Lambda^b_j &= \Lambda_j + \Lambda^b_{j+1} Q\left(\frac{\Lambda^f_j}{\Lambda^b_{j+1}}, N_j\right),
\end{aligned} \tag{A.12}$$

and, at the final step,

$$\Lambda^b_1 = \Lambda_1 + \Lambda^b_2 Q\left(\frac{\Lambda_1}{\Lambda^b_2}, N_1\right). \tag{A.13}$$

By repeating the process and constructing a new forward aggregation based on this backward aggregation, and so on, the following iterative algorithm is obtained:

$$\begin{aligned}
&\Lambda^f_i(s+1) = \Lambda_i + \Lambda^f_{i-1}(s+1) Q\left(\frac{\Lambda^b_i(s)}{\Lambda^f_{i-1}(s+1)}, N_{i-1}\right), \quad i = 2,\dots,M,\\
&\Lambda^b_j(s+1) = \Lambda_j + \Lambda^b_{j+1}(s+1) Q\left(\frac{\Lambda^f_j(s+1)}{\Lambda^b_{j+1}(s+1)}, N_j\right), \quad j = 1,\dots,M-1,\\
&s = 0,1,\dots, \quad \Lambda^b_i(0) = \Lambda_i, \quad \Lambda^f_1(s) = \Lambda_1, \quad \Lambda^b_M(s) = \Lambda_M, \quad \forall s.
\end{aligned} \tag{A.14}$$

Procedure (A.14) generates the following two sequences of throughput estimates:

$$\begin{aligned}
X^f_M(s) &= 1 - \Lambda^f_M(s)\varepsilon,\\
X^b_M(s) &= 1 - \Lambda^b_1(s)\varepsilon.
\end{aligned} \tag{A.15}$$

References

[1] G. Allon, T. Raviv, and R. Y. Rubinstein, *Application of the cross entropy method for buffer allocation problem in simulation based environment*, Proceedings of the Third Aegean International Conference on Design and Analysis of Manufacturing Systems (Tinos Island, Greece), 2001, pp. 269–278.

[2] T. Altiok, *Performance Analysis of Manufacturing Systems*, Springer-Verlag, New York, 1997.

[3] J. A. Buzacott and J. G. Shanthikumar, *Stochastic Models of Manufacturing Systems*, Prentice Hall, New Jersey, 1993.

[4] Y. Dallery, R. David, and X. Xie, *An efficient algorithm for analysis of transfer lines with unreliable machines and finite buffers*, IIE Trans. **20** (1988), no. 3, 280–283.

[5] Y. Dallery and S. B. Gershwin, *Manufacturing flow line systems: a review of models and analytical results*, Queueing Syst. Theory Appl. **12** (1992), no. 1-2, 3–94.

[6] S. B. Gershwin, *An efficient decomposition method for the approximate evaluation of tandem queues with finite storage space and blocking*, Oper. Res. **35** (1987), no. 2, 291–305.

[7] ———, *Manufacturing Systems Engineering*, Prentice Hall, New Jersey, 1994.

[8] S. B. Gershwin and J. E. Schor, *Efficient algorithms for buffer space allocation*, Ann. Oper. Res. **93** (2000), 117–144.
[9] C. Heavey, H. T. Papadopoulos, and J. Browne, *The throughput rate of multistation unreliable production lines*, Eur. J. Oper. Res. **68** (1993), no. 1, 69–89.
[10] S. Helber, *Performance Analysis of Flow Lines with Non-Linear Flow of Material*, Lecture Notes in Economics and Mathematical Systems, vol. 473, Springer-Verlag, Berlin, 1999.
[11] F. S. Hillier and R. W. Boling, *Finite queues in series, with exponential or Erlang service times—a numerical approach*, Oper. Res. **15** (1967), 286–303.
[12] M. A. Jafari and J. G. Shanthikumar, *Determination of optimal buffer storage capacities and optimal allocation in multistage automatic transfer lines*, IIE Trans. **21** (1989), no. 2, 130–135.
[13] P. Kubat and U. Sumita, *Buffers and backup machines in automatic transfer lines*, Int. J. Prod. Res. **23** (1985), no. 6, 1259–1270.
[14] J.-T. Lim, S. M. Meerkov, and F. Top, *Homogeneous, asymptotically reliable serial production lines: theory and a case study*, IEEE Trans. Automat. Control **35** (1990), no. 5, 524–534.
[15] C. T. Papadopoulos and T. I. Karagiannis, *A genetic algorithm approach for the buffer allocation problem in unreliable production lines*, International Journal of Operations and Quantitative Management **7** (2001), no. 1, 23–35.
[16] H. T. Papadopoulos and C. Heavey, *Queueing theory in manufacturing systems analysis and design: a classification of models for production and transfer lines*, Eur. J. Oper. Res. **92** (1996), no. 1, 1–27.
[17] H. T. Papadopoulos, C. Heavey, and J. Browne, *Queueing Theory in Manufacturing Systems Analysis and Design*, Chapman & Hall, London, 1993.
[18] H. T. Papadopoulos and M. I. Vidalis, *A heuristic algorithm for the buffer allocation in unreliable unbalanced production lines*, Computers & Industrial Engineering **41** (2001), no. 3, 261–277.
[19] S. G. Powell, *Buffer allocation in unbalanced three-station serial lines*, Int. J. Prod. Res. **32** (1994), no. 9, 2201–2217.
[20] B. A. Sevastyanov, *Influence of storage bin capacity on the average standstill time of a production line*, Theory Probab. Appl. **7** (1962), 429–438.
[21] D. D. Spinellis and C. T. Papadopoulos, *A simulated annealing approach for buffer allocation in reliable production lines*, Ann. Oper. Res. **93** (2000), 373–384.
[22] D. D. Spinellis, C. T. Papadopoulos, and J. MacGregor Smith, *Large production line optimization using simulated annealing*, Int. J. Prod. Res. **38** (2000), no. 3, 509–541.
[23] H. Yamashita and T. Altiok, *Buffer capacity allocation for a desired throughput in production lines*, Proceedings of the Samos International Workshop on Performance Evaluation and Optimization of Production Lines (Samos Island, Greece), 1997, pp. 1–24.

A. C. Diamantidis: Department of Product and Systems Design Engineering, University of the Aegean, Hermoupolis, Syros 84100, Greece
E-mail address: adiama@syros.aegean.gr

C. T. Papadopoulos: Department of Product and Systems Design Engineering, University of the Aegean, Hermoupolis, Syros 84100, Greece
E-mail address: hpap@aegean.gr

Mathematical Problems in Engineering

Special Issue on Space Dynamics

Call for Papers

Space dynamics is a very general title that can accommodate a long list of activities. This kind of research started with the study of the motion of the stars and the planets back to the origin of astronomy, and nowadays it has a large list of topics. It is possible to make a division in two main categories: astronomy and astrodynamics. By astronomy, we can relate topics that deal with the motion of the planets, natural satellites, comets, and so forth. Many important topics of research nowadays are related to those subjects. By astrodynamics, we mean topics related to spaceflight dynamics.

It means topics where a satellite, a rocket, or any kind of man-made object is travelling in space governed by the gravitational forces of celestial bodies and/or forces generated by propulsion systems that are available in those objects. Many topics are related to orbit determination, propagation, and orbital maneuvers related to those spacecrafts. Several other topics that are related to this subject are numerical methods, nonlinear dynamics, chaos, and control.

The main objective of this Special Issue is to publish topics that are under study in one of those lines. The idea is to get the most recent researches and published them in a very short time, so we can give a step in order to help scientists and engineers that work in this field to be aware of actual research. All the published papers have to be peer reviewed, but in a fast and accurate way so that the topics are not outdated by the large speed that the information flows nowadays.

Before submission authors should carefully read over the journal's Author Guidelines, which are located at http://www.hindawi.com/journals/mpe/guidelines.html. Prospective authors should submit an electronic copy of their complete manuscript through the journal Manuscript Tracking System at http://mts.hindawi.com/ according to the following timetable:

Manuscript Due	July 1, 2009
First Round of Reviews	October 1, 2009
Publication Date	January 1, 2010

Lead Guest Editor

Antonio F. Bertachini A. Prado, Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 12227-010 São Paulo, Brazil; prado@dem.inpe.br

Guest Editors

Maria Cecilia Zanardi, São Paulo State University (UNESP), Guaratinguetá, 12516-410 São Paulo, Brazil; cecilia@feg.unesp.br

Tadashi Yokoyama, Universidade Estadual Paulista (UNESP), Rio Claro, 13506-900 São Paulo, Brazil; tadashi@rc.unesp.br

Silvia Maria Giuliatti Winter, São Paulo State University (UNESP), Guaratinguetá, 12516-410 São Paulo, Brazil; silvia@feg.unesp.br