

Evaluative Models of Discrete Part Production Lines

The focus here is on discrete part production lines with asynchronous movement where each part produced is distinct. Production lines processing fluids and other continuous materials are not considered. From here on, when reference is made to production lines, discrete part production lines will be understood. In a production or flow line, all jobs are required to pass through each station in the same sequence once. These lines are usually associated with scale rather than scope, and a major advantage of production lines is the associated simple materials handling requirements.

A production line consists of work-stations, materials, human resources, and inter-work-station storage facilities. Storage facilities have a finite capacity. Randomness is introduced due to random processing times and the random behavior of work-stations in relation to failure and repair. In terms of classical queueing theory, production lines would be described as finite buffer tandem queueing systems where the work-stations are the servers, storage facilities are the buffers or the waiting lines, and the jobs are the customers.

In Figure 2.1, which depicts a K -work-station production line, $WS_i, i = 1, 2, \dots, K$ represents work-station i and $B_i, i = 1, 2, \dots, K$ denotes the buffer capacity of the buffer located in front of station WS_i . As there are K work-stations, there are $K - 1$ intermediate buffers. As described in Chapter 1, the goal of evaluative models is to calculate some performance measures of the system under study. The most usual performance measure determined is throughput. Each station may consist of a single perfectly reliable machine or an unreliable machine or a number of identical parallel reliable or unreliable machines. For notational purposes only, it should be understood that the word “machines” may cover operators.

In Section 2.1, Markovian analysis of production lines is presented using the underlying queueing system structure of production lines. It produces an exact analysis of such lines. In sub-section 2.1.1, a numerical approach is presented for solving the system of linear equations derived from the Markovian analysis. In sub-section 2.1.2, an algorithm is given for the generation of the conservative matrix A for the case of an exponential production line with inter-station buffers. In sub-section 2.1.3, a simple merge model of a two-station production line with merge operations (a non-linear model) is analyzed using exact Markovian methods.

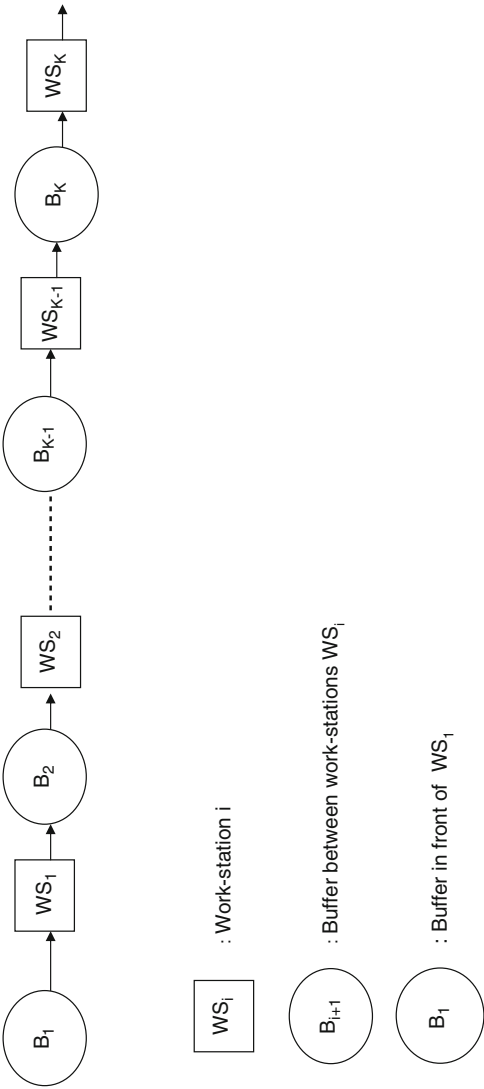


Fig. 2.1. A K -work-station production line

However, even for linear production lines with a large number of stations ($K > 6$) and reasonable buffer sizes, it is not possible to develop exact numerical results due to the complexity of the numerical calculations involved. As a result of this restriction, approximate solutions were sought. The decomposition approach is described in Section 2.2. Essentially, the process involves the decomposition of a large production line into a number of smaller lines with suitable provision for their inter-connection so that the behavior of the inter-connected system approximates the behavior of the original large production line.

Another approximation technique named the expansion method is given in Section 2.3.

Although the focus up to now has been on perfectly reliable machines at each station, it should be noted that the Markovian and decomposition methods can each handle unreliable machines. The aggregation method, which is a different approximation approach, was specifically developed to analyze transfer lines with asymptotically reliable stations. The aggregation method is covered in Section 2.4.

Up to this point the models used have been of a serial type, in the reliability sense, such that if a particular work-station was not operating due to breakdown or otherwise, the work-stations downstream from that particular work-station would eventually be starved. Work-stations in parallel were introduced with the result that the breakdown of a particular work-station would not necessarily lead to the starvation of stations downstream. The solution of production lines with parallel machines at each work-station is given in Section 2.5. The exact analysis of a simple parallel-machine production line consisting of two work-stations is presented in sub-section 2.5.1. An alternative exact analysis for solving the same two-station production line with parallel machines at each station which serves as building block in decomposing larger lines is given in sub-section 2.5.2. In sub-section 2.5.3 the approximate solution using decomposition method of large serial production lines with parallel-machine stations is given.

Simulation is often used when analytical methods prove intractable or are confined to rather simplified assumptions. In the case of production lines, simulation models may be used to assess the results of all approximate models and to obtain results using distributions for processing, failure, and repair times other than exponential or phase-type. Such models are explored in Section 2.6.

2.1 Markovian Model

Consider the model as depicted in Figure 2.1. Jobs enter station 1 from buffer B_1 of unrestricted capacity according to a Poisson distribution with arrival rate λ . Each job enters the line at station 1, passes through all stations in order and leaves the K^{th} station (last) in finished form. All jobs at each station are processed according to a First-In-First-Out (FIFO) queueing discipline.

The assumptions of the model are summarized below:

- (i) The processing or service times are exponentially or Erlang distributed random variables with mean rates equal to μ_i , $i = 1, 2, \dots, K$. In general, the service rates need not be identical (i.e., $\mu_i \neq \mu_j$ for $i \neq j$).
- (ii) All buffers between successive stations have finite capacities not necessarily of the same size.
- (iii) Blocking of a station occurs if the downstream buffer is full at the time of service completion.
- (iv) A station may be assumed to be perfectly reliable or subject to random failure according to an exponential distribution with mean rates equal to β_i , $i = 1, 2, \dots, K$. In general, the failure rates need not be identical (i.e., $\beta_i \neq \beta_j$ for $i \neq j$). However, it is assumed that a failure of a station can only occur when it is operating, i.e., operational-dependent breakdowns.
- (v) If a station fails, the part which the station was processing remains at the station, i.e., it is not placed in the preceding buffer.
- (vi) Once a failed station is repaired, it resumes processing at the same phase of service at which it failed, on the job that was not completed, and as a result of the memoryless property of the exponential distribution, the remaining processing time in that phase is exponentially distributed.
- (vii) The repair times are exponentially or Erlang distributed random variables with mean rates equal to r_i , $i = 1, 2, \dots, K$. In general, the repair rates need not be identical (i.e., $r_i \neq r_j$ for $i \neq j$).
- (viii) The general rule that deliberate idleness at a station is not allowed applies.
- (ix) A basic assumption is that the first station is never starved and the last station is never blocked. Although the arrival process is assumed to be Poisson, it is a necessary assumption of the model that the first station is never starved. This assumption characterizes the *saturated* line of the saturation model. The fact that the last station is never blocked relates to the storage capacity for final products.

The system under consideration is a two-dimensional stochastic process $N(t) = [N_1(t), N_2(t)]$. Both coordinate random variables are integer valued and nonnegative. $N_1(t)$ represents the number of jobs queued up in front of the first station at time t , and N_1 is the expected value of this quantity at equilibrium (the limit of $N_1(t)$, as t tends to infinity). There is no upper limit for N_1 . $N_2(t)$ represents the state of the sub-network at time t , which consists of stations $1, 2, 3, \dots, K$ and the intermediate buffers. In effect, $N_2(t)$ is a vector representing the situation in each station and in each of the intermediate buffers of the production line at time t . The number of states in the sub-network equals m , for some finite m . When $N_1 = 0$, the number of states in the sub-network equals m_0 , $m_0 < m$.

The changes in the state of the system are caused by the occurrence of various events. The occurrence times for all events have negative exponential or Erlang distributions with strictly positive means. Thus the process is Markovian. Its state-space is $S = \{(i, j) : i \geq 0, 1 \leq j \leq m\}$ with the index i specifying the total number of jobs queued up at the first station. Such customers are called “I-customers.” The index j

determines the state of the sub-network (Unit-II). It is important to note that **upon entering service at station 1, a customer becomes a “II-customer.”**

The transition matrix P that describes the model has the following block tri-diagonal form:

$$P = \begin{vmatrix} A_{01} & A_0 & & & & \\ A_2 & A_1 & A_0 & & & \\ & A_2 & A_1 & A_0 & & \\ & & A_2 & A_1 & A_0 & \\ & & & \ddots & \ddots & \ddots \\ & & & & \ddots & \ddots & \ddots \end{vmatrix} \quad (2.1)$$

and the equilibrium equations $\pi P = 0$ can be expressed in matrix-difference form as

$$\pi_k A_0 + \pi_{k+1} A_1 + \pi_{k+2} A_2 = 0 \quad (2.2)$$

for $k = 0, 1, 2, \dots$ and

$$\pi_0 A_{01} + \pi_1 A_2 = 0 \quad (2.3)$$

for the boundary equations, where,

A_0 is an $(m \times m)$ -matrix describing the transitions in the sub-network, which simultaneously produce inputs to the first queue.

A_1 is an $(m \times m)$ -matrix describing transitions in the sub-network which produce neither inputs to nor outputs from the first queue assuming that the queue is *not empty*.

A_2 is an $(m \times m)$ -matrix describing transitions in the sub-network which simultaneously produce outputs from the first queue, and

A_{01} is an $(m \times m)$ -matrix describing transitions in the sub-network which produce neither inputs to nor outputs from the first queue, assuming that the queue is *empty*.

A Markov chain whose equilibrium equations have the form of equations (2.2) and (2.3) is known as a Quasi-Birth and Death (QBD) process.

Let P_2 be the steady-state probabilities of the sub-network, assuming that the first queue is never empty. Solving the equations

$$\begin{aligned} P_2 A &= 0 \\ P_2 \mathbf{e} &= 1 \end{aligned} \quad (2.4)$$

where A is the conservative stable matrix given by

$$A = A_0 + A_1 + A_2 \quad (2.5)$$

and \mathbf{e} is an $(m \times 1)$ column-vector, with all elements equal to 1, will give explicit results for $P_2(j)$, $j = 1, 2, \dots, m$.

The equilibrium condition is given by

$$P_2 A_2 \mathbf{e} > P_2 A_0 \mathbf{e}. \quad (2.6)$$

Table 2.1. Notation

<i>Symbol</i>	<i>Meaning</i>
K	Number of stations
B_i	Buffer capacity preceding the i^{th} station. Note: when $B_i = B_j$ for all i , then the buffer capacity is denoted by B
n_i	Status of buffer i
s_i	Status of station i
P_i	Denotes the number of phases of the service (processing) distribution of the i^{th} station
R_i	Denotes the number of phases of the repair distribution of the i^{th} station
$m_{K,P}^{B,R}$	Number of states in the sub-network with K stations, each buffer having the same capacity B , each service distribution having P phases and each repair distribution having R phases
$m_{K,P_1,P_2,\dots,P_K}^{B_2,\dots,B_K,R_1,R_2,\dots,R_K}$	Number of states in the sub-network of a K station system with buffer capacities B_2, \dots, B_K . The number of phases of each station's service distribution is equal to P_1, P_2, \dots, P_K phases and the number of phases of each station's repair distribution is equal to R_1, R_2, \dots, R_K

From this relationship the critical mean input rate (λ^*) to the system can be determined. In the steady-state, this critical input rate is identical to the maximum throughput rate of the production line. By calculating the throughput of the system as outlined above, we exclude the states of the system where the first station is empty, i.e., sub-matrix A_{01} is not included. Therefore, the throughput of the system is governed by the assumption that the first queue is never empty (saturation model).

The notation used is shown in Table 2.1.

The states of the sub-network are described by the following vector:

$$(s_1, n_2, s_2, n_3, \dots, n_K, s_K) \quad (2.7)$$

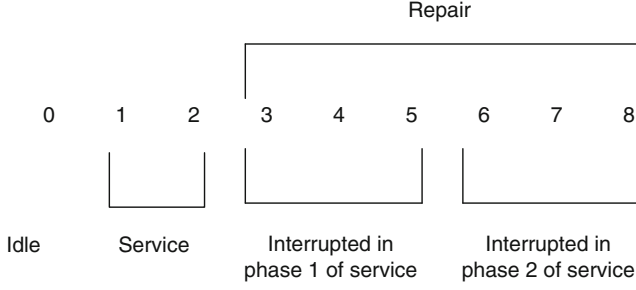
where, s_i can take any value from 0 to $(P_i + R_i \times P_i)$:

$s_i = 0$ – station is idle,

$s_i = 1, \dots, P_i$ – station is in service,

$s_i = (P_i + 1), \dots, (P_i + P_i \times R_i)$ – station is in repair.

After a station has been repaired, it is assumed that service is resumed at the phase in which the station was interrupted. Therefore, there is a need to keep a record of the phase of service in which the station was interrupted. It should be noted that the need to account for the phase is a modeling requirement and may not have any



Total number of states = $P_i(1 + R_i) = 8$

Fig. 2.2. The states of s_i for $P_i = 2, R_i = 3$

corresponding physical meaning. This results in using $(P_i + P_i \times R_i)$ states to describe a station's repair process. It also necessitates the use of equation (2.8) to transfer s_i from a state in service to a state in repair and equation (2.9) to do the reverse.

$$\text{Beginning repair state} = (P_i + ((s_i - 1) \times R_i) + 1) \quad (2.8)$$

$$\text{Phase to resume service at} = \frac{(s_i - P_i)}{R_i}. \quad (2.9)$$

The states that s_i can take, for the parameters $P_i = 2$ and $R_i = 3$, are illustrated in Figure 2.2.

n_i can take values from 0 to $(B_i + 1)$. The values from 0 to B_i denote the number of items in buffer B_i with B_i also denoting the capacity of buffer B_i . When $n_i = (B_i + 1)$, station $(i - 1)$ is blocked.

The following recursive relationship was obtained (in Heavey, Papadopoulos and Browne, 1993) to calculate the number of, states of a system with K stations with parameters $P_i, R_i, B_j, i = 1, \dots, K, j = 2, \dots, K$.

The number of states for a two-station system is first calculated using the parameters $P_{K-1}, P_K, R_{K-1}, R_K, B_K$.

Two-station system:

$$\Xi_1 = (P_K + (P_K \times R_K) + 1) \quad (2.10)$$

$$\Xi_2 = (((P_{K-1} \times (B_K + 1)) + (P_{K-1} \times R_{K-1} \times (B_K + 1))) + 1) \quad (2.11)$$

$$\Xi_3 = ((B_K \times (P_{K-1} + (P_{K-1} \times R_{K-1}))) + 1) \quad (2.12)$$

$$\Omega_2 = ((\Xi_1 \times \Xi_2) - \Xi_3). \quad (2.13)$$

Ω_2 will equal the number of states of the system if $K = 2$. To calculate the number of states for systems with $K > 2$, the following recursive scheme is used. Before entering the loop below, the variable Ω_1 is set equal to $(P_K + P_K \times R_K)$ and the variable

Table 2.2. Number of states for $P = 1, R = 1$ and identical buffer capacities

# of Stations	Buffer Size				
	0	1	2	3	4
2	8	12	16	20	24
3	30	70	126	198	286
4	112	408	992	1,960	3,408
5	418	2,378	7,810	19,402	40,610
6	1,560	13,860	61,488	192,060	483,912
7	5,822	80,782	484,094	1,901,198	5,766,334
8	21,728	470,832	3,811,264	18,819,920	68,712,096
9	81,090	2,744,210	30,006,018	188,119,920	818,778,818

Ω_2 takes its value from equation (2.13), with the parameters $P_{K-1}, P_K, R_{K-1}, R_K, B_K$ of the K -station system used in equations (2.10), (2.11) and (2.12).

DO $I = (K - 1)$ to 2, -1

$$Y_1 = ((\Omega_2 - \Omega_1)/(P_I \times (R_I + 1))) + \Omega_1 \quad (2.14)$$

$Y_2 :=$ Using parameters N_I, P_I, P_{I-1} ,

R_I, R_{I-1} calculate the number of states

for a two-station system as above, i.e., let

$$B_K = N_I, P_{K-1} = P_{I-1}, P_K = P_I,$$

$R_{K-1} = R_{I-1}, R_K = R_I$ in equations

$$(2.10), (2.11), (2.12) \text{ above.} \quad (2.15)$$

$$Y_{31} = (P_{I-1} + P_{I-1} \times R_{I-1})$$

$$Y_{32} = (P_I + P_I \times R_I)$$

$$Y_{33} = (B_I \times (Y_{31} \times (Y_{32} - 1)))$$

$$Y_3 = ((Y_{31} \times Y_{32}) + (Y_{32} - 1) + Y_{33}) \times \Omega_1 \quad (2.16)$$

$$\Omega_1 = \Omega_2$$

$$\Omega_2 = ((Y_1 \times Y_2) - Y_3)$$

END DO I

$$m_{K, P_1, \dots, P_K}^{B_2, \dots, B_K, R_1, R_2, \dots, R_K} = \Omega_2 \quad (2.17)$$

Table 2.2 gives the number of states for K -station systems, with identical buffers equal to B , $P = 1$ and $R = 1$. As one can observe from Table 2.2, the number of states becomes very large, even for relatively small systems.

2.1.1 A numerical approach

As is well known, there are a number of ways of solving sets of homogeneous linear equations. To name a few, Gaussian elimination, iterative methods based on

Gauss-Siedel approximation, Jacobian elimination, and matrix recursive methods. In the solution of such sets of homogeneous linear equations, the analyst is primarily concerned with efficiency of calculations and rapidity of convergence and estimation of the degree of approximation, if appropriate. Clearly, because of the number of states, in any realistic model of a production line, there is a need for an efficient algorithm to determine the steady-state probabilities associated with the states of the system.

The algorithm outlined below is based on Gaussian elimination with a dynamically adjusted successive over-relaxation factor to achieve rapid convergence. The essential components of this algorithm are

- Ordering of the states
- Generation of the transition matrix
- Solution of the resulting system of linear equations

This algorithm was coded in C++ by Dr. Cathal Heavey and is based on the work of Heavey, Papadopoulos and Browne (1993), Papadopoulos, Heavey and O’Kelly (1989, 1990), Papadopoulos and O’Kelly (1989) and Papadopoulos (1989) and with appropriate instructions is available at the website associated with this book with the abbreviated name MARKOV. The user inputs into the algorithm the following parameters: K the number of stations; B_2, \dots, B_K the buffer capacities; P_1, \dots, P_K the number of phases of the service distribution for each station; R_1, \dots, R_K , the number of phases of the repair distribution for each station; μ_i the mean service rates; r_i the mean repair rates; and β_i the mean breakdown rates. Thus, a very general algorithm has been developed which generates the transition probability matrix, A , and then solves the set of linear equations via the use of the SOR method and gives as output the throughput, X_K , of a K -station production line with finite intermediate buffers and with the service and repair times following a phase-type distribution and the times to failure being exponentially distributed.

The ordering of the states affects the structure of the conservative matrix A . The objective is to find an ordering of the states such that matrix A will have as simple a structure as possible from a computational point of view. This will facilitate the development of a very efficient algorithm for the generation of matrix A . To select an appropriate ordering of the states, a criterion for the structure of matrix A must be selected. In the algorithm included at the website associated with this book, the criterion used was to keep the non-zero elements of the conservative matrix A as close as possible to the diagonal elements, i.e., a quasi band diagonal matrix. Because of the increasing number of states, as system complexity increases, it is not possible to assess how close matrix A is to a strict band diagonal matrix.

A recursive algorithm for generating the conservative matrix A has been developed based on the generation of a series of sub-matrices (Heavey, Papadopoulos and Browne, 1993). Specific details of the matrix generation process for the case of a reliable exponential production line with inter-station buffers (Papadopoulos, Heavey and O’Kelly, 1989) are given in sub-section 2.1.2.

Table 2.3. Exponential service, repair, and failure, $K = 3$

$B_2 = 2, B_3 = 4$ $\mu_1 = 1.5, \mu_2 = 2.0, \mu_3 = 1.9$ $r_1 = 0.1, r_2 = 0.02, r_3 = 0.15$ $\beta_1 = 0.02, \beta_2 = 0.01, \beta_3 = 0.09$	
Analytical Results	Simulation Results 95% CI
$X_3 = 0.7346$	$0.721 - 0.737 - 0.752$

$B_2 = 5, B_3 = 3$ $\mu_1 = 2.6, \mu_2 = 3.0, \mu_3 = 3.2$ $r_1 = 0.5, r_2 = 0.03, r_3 = 0.15$ $\beta_1 = 0.03, \beta_2 = 0.01, \beta_3 = 0.02$	
Analytical Results	Simulation Results 95% CI
$X_3 = 1.2985$	$1.26 - 1.28 - 1.31$

An iterative method was used to solve the system of linear equations. The iterative method used was the Successive Over Relaxation (SOR) method. SOR is more efficient than the Gauss-Seidel method, but SOR has one main drawback, the unknown optimal value of the relaxation factor. A process of dynamically adjusting the relaxation factor has been introduced into the algorithm, which worked well in practice.

The results of the algorithm have been compared with available analytical results (systems with a small number of states) and simulation studies on systems with relatively large number of states and has been found to be satisfactory.

A sample of the throughput rate, X_K , from the analytical model, compared with results from a simulation model are given below. Two arbitrary examples are given for systems with: (1) Exponential service, repair, and failure (see Table 2.3); (2) Erlang service, exponential repair, and failure (see Table 2.4); (3) Erlang service, repair, and exponential failure (see Table 2.5). $\mu_i, r_i, \beta_i, i = 1, 2, 3$ are the mean service rates, repair rates, and failure rates, respectively.

As can be seen from Tables 2.3, 2.4 and 2.5, the point estimates of the throughput from the simulation model are very close to the results from the analytical model and all the analytical results are covered by the 95% confidence intervals (CI). These results are typical of all the models tested against simulation. Therefore, it can be safely concluded that the analytical model yields the correct results.

In sub-section 2.1.2, the detailed development of the conservative matrix A for the reliable exponential case is developed. Details of more general cases (phase-type distribution of service and repair times) are available in the literature listed at the end of this chapter.

Table 2.4. Erlang service, exponential repair, and failure, $K = 4$

$P_1 = 3, P_2 = 2, P_3 = 3, P_4 = 2$ $B_2 = 3, B_3 = 2, B_4 = 3$ $\mu_1 = 5.0, \mu_2 = 4.5, \mu_3 = 5.2, \mu_4 = 3.7$ $r_1 = 0.05, r_2 = 0.03, r_3 = 0.07, r_4 = 0.1$ $\beta_1 = 0.02, \beta_2 = 0.001, \beta_3 = 0.003, \beta_4 = 0.05$	
Analytical Results	Simulation Results 95% CI
$X_4 = 2.0025$	$1.99 - 2.04 - 2.10$

$P_1 = 2, P_2 = 3, P_3 = 3, P_4 = 2$ $B_2 = 3, B_3 = 5, B_4 = 3$ $\mu_1 = 1.5, \mu_2 = 0.9, \mu_3 = 0.9, \mu_4 = 1.5$ $r_1 = 0.1, r_2 = 0.03, r_3 = 0.2, r_4 = 0.3$ $\beta_1 = 0.02, \beta_2 = 0.01, \beta_3 = 0.09, \beta_4 = 0.2$	
Analytical Results	Simulation Results 95% CI
$X_4 = 0.4591$	$0.450 - 0.456 - 0.462$

Table 2.5. Erlang service, repair, and exponential failure, $K = 4$

$P_1 = 3, P_2 = 2, P_3 = 3, P_4 = 2$ $R_1 = 2, R_2 = 3, R_3 = 4, R_4 = 2$ $B_2 = 2, B_3 = 1, B_4 = 3$ $\mu_1 = 2.5, \mu_2 = 1.9, \mu_3 = 2.6, \mu_4 = 3.0$ $r_1 = 0.05, r_2 = 0.03, r_3 = 0.07, r_4 = 0.1$ $\beta_1 = 0.02, \beta_2 = 0.001, \beta_3 = 0.003, \beta_4 = 0.05$	
Analytical Results	Simulation Results 95% CI
$X_4 = 1.1325$	$1.10 - 1.12 - 1.14$

$P_1 = 2, P_2 = 3, P_3 = 3, P_4 = 2$ $R_1 = 3, R_2 = 2, R_3 = 3, R_4 = 2$ $B_2 = 1, B_3 = 2, B_4 = 3$ $\mu_1 = 5.0, \mu_2 = 4.5, \mu_3 = 5.2, \mu_4 = 3.7$ $r_1 = 0.1, r_2 = 0.03, r_3 = 0.2, r_4 = 0.3$ $\beta_1 = 0.02, \beta_2 = 0.01, \beta_3 = 0.09, \beta_4 = 0.2$	
Analytical Results	Simulation Results 95% CI
$X_4 = 1.5958$	$1.53 - 1.54 - 1.60$

Table 2.6. Notation

<i>Symbol</i>	<i>Meaning</i>
K	Number of stations.
B_i	Buffer capacity preceding the i^{th} station. Note: when $B_i = B_j$ for all i , then the buffer capacity is denoted by B .
n_i	Status of buffer i .
s_i	Status of station i (see Table 2.7).
m_K^B	Number of states in the sub-network of a K station system with identical buffers, each of capacity B .
$m_K^{B_2, \dots, B_K}$	Number of states in the sub-network of a K station system with non-identical buffers, with buffer capacities B_2, \dots, B_K .

Table 2.7. States of station i

s_i	<i>Meaning</i>
0	Station is idle.
1	Station is busy.
2	Station is busy and blocking preceding station.

2.1.2 The algorithm for the generation of the conservative matrix A for the reliable exponential production lines with inter-station buffers

For purposes of illustration, in this sub-section, the recursive algorithm will be applied to the case of a reliable exponential production line only (see Papadopoulos, Heavey and O’Kelly, 1989). Table 2.6 lists the notation used in this sub-section.

The algorithm for generating the conservative matrix A is divided into two parts. The first part generates sub-matrix $Y1_{K=k}^*$ for the appropriate system. The second part generates sub-matrix $Y2_{K=k}^*$ from sub-matrix $Y1_{K=k}^*$, and the non-diagonal elements $P_1\mu_1, R_1r_1$ and β_1 . In the first part of the algorithm, the non-zero elements of $Y1_{K=k}^*$, the column coefficients, and the number of elements in each row are stored in separate one dimensional arrays.

The second part of the algorithm is executed during the execution of the solution procedure. As a consequence, the non-zero elements of sub-matrix $Y2_{K=k}^*$, and the non-diagonal elements $P_1\mu_1, R_1r_1$ and β_1 need not be stored in memory. This can greatly reduce the amount of memory required to solve a system.

The states of the sub-network are described by the following vector:

$$(n_2, s_2, n_3, s_3, \dots, n_K, s_K) \quad (2.18)$$

s_1 is not included in the state vector because it is always equal to 1, i.e., the first station is never idle. This does not mean that the first station cannot be blocked by buffer B_2 or work-station WS_2 . s_i can take any of the values listed in Table 2.7, and n_i can take any value from 0 to B_i , as it denotes the number of items in buffer i .

The set of linear equations for the solution of P_2 , the marginal p.d.f. for the sub-network, can be written in the following two ways.

$$P_2 A = 0 \quad (2.19)$$

$$A^T P_2 = 0. \quad (2.20)$$

In the rest of this sub-section, A^T is examined. This is because in order to generate matrix A efficiently, the relationship between its columns (rows of A^T) needs to be examined. In order to simplify the notation, A denotes A^T in the rest of this sub-section.

Number of States

A prerequisite to the development of the algorithm is the derivation of an equation to calculate the number of states in the sub-network. The case where buffers are identical is investigated first and then the case of buffers being non-identical.

Identical Buffers

For this case, where buffers are of equal capacity, say $B = N$, the following difference equation is obtained, in a way analogous to that used for the case where buffers were not allowed (see Papadopoulos, 1989 and Papadopoulos and O'Kelly, 1989):

$$m_{K+2}^N - (N+3)m_{K+1}^N + m_K^N = 0. \quad (2.21)$$

Then, its *characteristic equation* is

$$x^2 - (N+3)x + 1 = 0,$$

with two real roots:

$$x_1 = \frac{(N+3) + \sqrt{(N+3)^2 - 4}}{2}, \quad x_2 = \frac{(N+3) - \sqrt{(N+3)^2 - 4}}{2}.$$

Therefore the general solution of (2.21) is

$$\begin{aligned} m_K^N &= c_1 x_1^K + c_2 x_2^K \\ &= c_1 \left(\frac{(N+3) + \sqrt{(N+3)^2 - 4}}{2} \right)^K + c_2 \left(\frac{(N+3) - \sqrt{(N+3)^2 - 4}}{2} \right)^K. \end{aligned}$$

The initial conditions: $m_0 = 0$ and $m_1 = 1$ give

$$c_1 + c_2 = 0,$$

and

$$c_1 \left(\frac{(N+3) + \sqrt{(N+3)^2 - 4}}{2} \right) + c_2 \left(\frac{(N+3) - \sqrt{(N+3)^2 - 4}}{2} \right) = 1.$$

Hence,

$$c_1 = -c_2 = \frac{1}{\sqrt{(N+3)^2 - 4}} = \frac{\sqrt{(N+3)^2 - 4}}{(N+3)^2 - 4},$$

and the general solution becomes

$$m_K^N = \left\{ \left(\frac{(N+3) + \sqrt{(N+3)^2 - 4}}{2} \right)^K - \left(\frac{(N+3) - \sqrt{(N+3)^2 - 4}}{2} \right)^K \right\} \left(\frac{1}{\sqrt{(N+3)^2 - 4}} \right). \quad (2.22)$$

Equation (2.22) was used to calculate the number of states for the systems in Table 2.8. It is clear from Table 2.8 that the number of states increases tremendously with an increase in the size of the buffer and in the number of stations. This places strict limits on the size of the system for which exact results can be obtained.

Non-identical Buffers

For this case, where buffers are of unequal capacity, say B_2, B_3, \dots, B_K , the difference equation may be shown to be similar to that obtained for Case 1, where buffers were of equal capacity (equation (2.21)), i.e.,

$$m_{K+2}^{B_2, B_3, \dots, B_{K+2}} = (B_{K+2} + 3) m_{K+1}^{B_2, B_3, \dots, B_{K+1}} - m_K^{B_2, B_3, \dots, B_K}. \quad (2.23)$$

Applying the initial conditions $m_0 = 0$ and $m_1 = 1$ to equation (2.23), for $K = 0, 1, \dots$, sequentially,

(1) $K = 0$:

$$\begin{aligned} m_2^{B_2} &= (B_2 + 3) m_1 - m_0 \\ &= (B_2 + 3)(1) - 0 \\ &= B_2 + 3. \end{aligned} \quad (2.24)$$

(2) $K = 1$: Combination of equations (2.23) and (2.24) gives

$$\begin{aligned} m_3^{B_2, B_3} &= (B_3 + 3) m_2^{B_2} - m_1 \\ &= (B_2 + 3)(B_3 + 3) - 1. \end{aligned} \quad (2.25)$$

(3) $K = 2$: Combination of equations (2.23), (2.24) and (2.25) gives

$$\begin{aligned} m_4^{B_2, B_3, B_4} &= (B_4 + 3) m_3^{B_2, B_3} - m_2^{B_2} \\ &= (B_4 + 3) [(B_3 + 3)(B_2 + 3) - 1] - (B_2 + 3) \\ &= (B_2 + 3) [(B_3 + 3)(B_4 + 3) - 1] - (B_4 + 3). \end{aligned} \quad (2.26)$$

Table 2.8. Number of states of the system

[illegible]

(4) $K = 3$: Combination of equations (2.23), (2.25) and (2.26) gives

$$\begin{aligned}
 m_5^{B_2, B_3, B_4, B_5} &= (B_5 + 3) m_4^{B_2, B_3, B_4} - m_3^{B_2, B_3} \\
 &= (B_5 + 3) \{ (B_4 + 3) [(B_3 + 3) (B_2 + 3) - 1] - (B_2 + 3) \} \\
 &\quad - [(B_3 + 3) (B_2 + 3) - 1] \\
 &= [(B_2 + 3) (B_3 + 3) - 1] [(B_4 + 3) (B_5 + 3) - 1] \\
 &\quad - (B_2 + 3) (B_5 + 3).
 \end{aligned} \tag{2.27}$$

(5) $K = 4$: Combination of equations (2.23), (2.26) and (2.27) gives

$$\begin{aligned}
 m_6^{B_2, \dots, B_6} &= (B_6 + 3) m_5^{B_2, \dots, B_5} - m_4^{B_2, \dots, B_4} \\
 &= (B_6 + 3) \{ [(B_2 + 3) (B_3 + 3) - 1] [(B_2 + 3) (B_3 + 3) - 1] \\
 &\quad - (B_2 + 3) (B_5 + 3) \} \\
 &\quad - \{ (B_2 + 3) [(B_3 + 3) (B_4 + 3) - 1] - (B_5 + 3) \},
 \end{aligned}$$

and after some algebra,

$$\begin{aligned}
 m_6^{B_2, \dots, B_6} &= (B_4 + 3) [(B_2 + 3) (B_3 + 3) - 1] [(B_5 + 3) (B_6 + 3) - 1] \\
 &\quad - (B_6 + 3) [(B_2 + 3) (B_3 + 3) - 1] \\
 &\quad - (B_2 + 3) [(B_5 + 3) (B_6 + 3) - 1].
 \end{aligned} \tag{2.28}$$

The examples illustrated above suggest the following iterative scheme to calculate the number of states of a system with non-identical buffers, i.e., a system with K stations and buffer capacities B_2, B_3, \dots, B_K .

Initial Values:

$$\begin{aligned}
 V1 &= 1 = m_1 \\
 V2 &= 0 = m_0 \\
 \text{DO } J &= 2 \text{ to } K \\
 &\quad V = (B_J + 3) V1 - V2 \\
 &\quad V2 = V1 \\
 &\quad V1 = V \\
 \text{END DO } J \\
 m_K^{B_2, \dots, B_K} &= V.
 \end{aligned}$$

The iterative scheme above calculates the number of states of a K station system with buffer capacities B_2, B_3, \dots, B_K , by first calculating $m_2^{B_2}$ and then $m_3^{B_2, B_3}$, i.e., by using B_i in the following order, $i = 2, 3, \dots, K - 1, K$. It is interesting to note that the number of states for a system with non-identical buffers can also be calculated using B_i in the reverse order, $i = K, K - 1, \dots, 2$, i.e., calculate $m_2^{B_K}$ first, then $m_3^{B_{K-1}, B_K}$ and so on. In the algorithm for the generation of the transition matrix, the latter method is used.

Ordering of States

Each state is represented by the following vector:

$$(n_2, s_2, n_3, \dots, n_K, s_K). \quad (2.29)$$

Each state is altered by the following rule:

If s_i equals 2 and $i > 2$ then

$$s_{i-1}^{altered} = (s_{i-1} - 1).$$

Then the ‘altered states’ are given a unique numerical value in order to ensure a 1 – 1 correspondence, as follows:

$$n_2 \times L^{E-1} + s_2^{altered} \times L^{E-2} + \dots + n_K \times L^{E-(E-1)} + s_K^{altered} \times L^{E-E} = \text{numerical value}$$

with E equal to the number of elements in the state vector and L given an appropriate integer value as follows:

$$L > \text{MAX}\{B_j, 2\}, \quad j = 2, \dots, K.$$

L is the base for the numerical values of the states. The numerical values of the ‘altered states’ are then ordered in increasing value and the states ordered according to this.

The above procedure will be illustrated with an example. Table 2.9 lists the states, the ‘altered states’, and the numerical values of the ‘altered states’ for $K = 3$, $B_2 = 0, B_3 = 1$. E equals 4 and L equals 3.

Only states (0,1,1,2) and (0,2,1,2) were altered (see Table 2.9). Table 2.10 gives the numerical values of the ‘altered states’ ordered in increasing value and the states ordered according to this ordering.

The reason for ordering the states is to give matrix A a relatively simple structure which can be exploited when developing the algorithm to generate matrix A . Matrix

Table 2.9. Altered states and their numerical values

States	Altered States	Numerical Value
(0,0,0,0)	(0,0,0,0)	0
(0,0,0,1)	(0,0,0,1)	1
(0,1,0,0)	(0,1,0,0)	9
(0,1,0,1)	(0,1,0,1)	10
(0,2,0,0)	(0,2,0,0)	18
(0,2,0,1)	(0,2,0,1)	19
(0,0,1,1)	(0,0,1,1)	4
(0,1,1,1)	(0,1,1,1)	13
(0,1,1,2)	(0,0,1,2)	5
(0,2,1,1)	(0,2,1,1)	22
(0,2,1,2)	(0,1,1,2)	14

Table 2.10. Ordering of states

Ordered States	Numerical Value
(0,0,0,0)	0
(0,0,0,1)	1
(0,0,1,1)	4
(0,1,1,2)	5
(0,1,0,0)	9
(0,1,0,1)	10
(0,1,1,1)	13
(0,2,1,2)	14
(0,2,0,0)	18
(0,2,0,1)	19
(0,2,1,1)	22

(2.30) gives matrix A for $K = 3, B_2 = 0, B_3 = 1$, with the states ordered according to Table 2.10. Note that each of the non-diagonal elements, μ_3 , μ_2 and μ_1 , in matrix (2.30) are always found in the same position relative to the diagonal element, i.e., μ_2 is always two columns to the right of the diagonal element.

$$A = \begin{pmatrix} -\mu_1 & \mu_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\mu_1 - \mu_3 & \mu_3 & 0 & \mu_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\mu_1 - \mu_3 & \mu_3 & 0 & \mu_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\mu_1 - \mu_3 & 0 & 0 & \mu_2 & 0 & 0 & 0 & 0 \\ \mu_1 & 0 & 0 & 0 & -\mu_1 - \mu_2 & \mu_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & \mu_1 & 0 & 0 & 0 & -\sum_{i=1}^3 \mu_i & \mu_3 & 0 & \mu_2 & 0 & 0 \\ 0 & 0 & \mu_1 & 0 & 0 & 0 & -\sum_{i=1}^3 \mu_i & \mu_3 & 0 & \mu_2 & 0 \\ 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & -\mu_3 & 0 & 0 & \mu_2 \\ 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & -\mu_2 & \mu_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & -\mu_2 - \mu_3 & \mu_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & 0 & -\mu_2 - \mu_3 \end{pmatrix} \quad (2.30)$$

Structure of Matrix A

Matrix A equals the summation of sub-matrices A_1 , A_2 and A_0 . Sub-matrices A_0 and A_2 have very simple structures whereas sub-matrix A_1 has a relatively complicated structure. Sub-matrix A_1 is examined first and then A_0 and A_2 .

Description of A_1

Matrix A_1 for any value K ($K > 2$) with identical or non-identical buffers was found to take the form described in Figure 2.3.

C , D , and D^* are all

$$m_{K-1}^{B_3, B_4, \dots, B_K} \times m_{K-1}^{B_3, B_4, \dots, B_K}$$

matrices. E and F are

$$\left(m_{K-1}^{B_3, B_4, \dots, B_K} - m_{K-2}^{B_4, \dots, B_K} \right) \times \left(m_{K-1}^{B_3, B_4, \dots, B_K} - m_{K-2}^{B_4, \dots, B_K} \right)$$

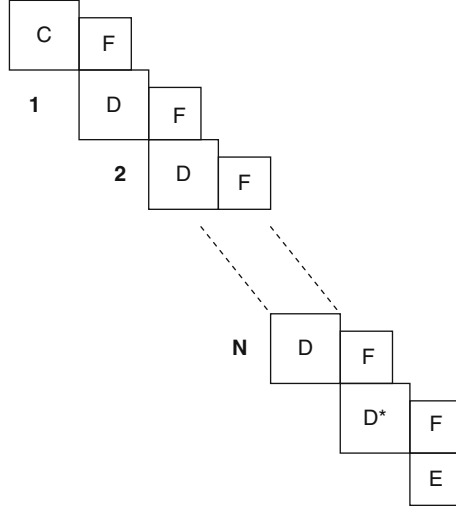


Fig. 2.3. Structure of A_1 , $K > 2$, $B_2 = N$, B_3, B_4, \dots, B_K

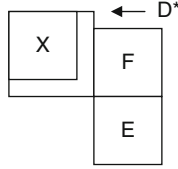


Fig. 2.4. Relationship of sub-matrix E to D^*

matrices. The number of times sub-matrices D and F appear between sub-matrices C and D^* equals $B_2 = N$. The relationships between the sub-matrices are as follows:

1. Sub-matrix C for a K station system with $B_2 = N, B_3, B_4, \dots, B_K$ is generated from A_1 for $K - 1$ station system with B_3, B_4, \dots, B_K , by: (i) Substituting μ_{i+1} for μ_i , $i = K, K - 1, \dots, 2$ (i.e., backwards) in $(A_1)_{K-1}$; (ii) Subtracting μ_1 from the last $m_{K-2}^{B_4, \dots, B_K}$ diagonal elements of $(A_1)_{K-1}$.
2. (i) D is generated from C by subtracting μ_2 from the first $(m_{K-1}^{B_3, B_4, \dots, B_K} - m_{K-2}^{B_4, \dots, B_K})$ diagonal elements of C .
 (ii) If $B_2 = 0$, then there is no sub-matrix D . Therefore D^* is generated from C . If $B_2 = 0$, then μ_1 is also added to the last $m_{K-2}^{B_4, \dots, B_K}$ diagonal elements of C .
3. D^* is generated from D by adding μ_1 to the last $m_{K-2}^{B_4, \dots, B_K}$ diagonal elements of D . If $B_2 = 0$, this relationship does not hold because there will be no sub-matrix D .
4. E is a $(m_{K-1}^{B_3, B_4, \dots, B_K} - m_{K-2}^{B_4, \dots, B_K}) \times (m_{K-1}^{B_3, B_4, \dots, B_K} - m_{K-2}^{B_4, \dots, B_K})$ matrix which is generated from X , a sub-matrix of D^* (see Figure 2.4), by adding μ_1 to all the diagonal elements of X .

5. F is a square diagonal matrix of order $\left(m_{K-1}^{B_3, B_4, \dots, B_K} - m_{K-2}^{B_4, \dots, B_K}\right)$ with μ_2 in the diagonal elements. The first sub-matrix F is positioned on the $\left(m_{K-2}^{B_4, \dots, B_K} + 1\right)$ row and the $\left(m_{K-1}^{B_3, B_4, \dots, B_K} + 1\right)$ column of matrix A_1 . Its position relative to D and D^* is the same as its position relative to C .

Once A_1 for $K = 2$, B_K is obtained, using the relationships outlined above, A_1 for any value $K, B_2 = N, B_3, B_4, \dots, B_K$ can be generated. A_1 for $K = 2$ and any value B_K is easy to generate.

Description of A_0 and A_2

In general A_0 is a $\left(m_K^{B_2, B_3, \dots, B_K} \times m_K^{B_2, B_3, \dots, B_K}\right)$ matrix with λ in all the diagonal elements and $\mu_K \theta$ in exactly the same positions as $\mu_K \theta'$ is in A_1 .

In general A_2 is a $\left(m_K^{B_2, B_3, \dots, B_K} \times m_K^{B_2, B_3, \dots, B_K}\right)$ matrix with μ_1 in the I^{th} column and the $\left(m_{K-1}^B + I\right)$ row with $I = 1, 2, \dots, \left(m_K^B - m_{K-1}^B\right)$.

Therefore the basic structure of $A = A_0 + A_1 + A_2$ is given by the structure of sub-matrix A_1 except:

1. A does not contain any λ , i.e., λ in the diagonal elements of A_0 cancels $-\lambda$ in the diagonal elements of A_1 .
2. Instead of $\mu_K \theta'$ in A_1 , there is a μ_K in A , i.e., $(\theta + \theta') = 1$. This is because $\mu_K \theta$ in A_0 is in exactly the same position as $\mu_K \theta'$ is in A_1 .
3. The inclusion of the sub-matrix A_2 .

Figure 2.5 gives the structure of sub-matrix A for a K station system ($K > 2$). Sub-matrices C, D, D^* , and E are as described in sub-section 2.1.2 except the changes outlined above, i.e., their diagonal elements do not contain any λ and $\mu_K \theta' \rightarrow \mu_K$. Sub-matrices G and H contain the μ_1 elements of A_2 . G is a square matrix of order $m_{K-1}^{B_3, B_4, \dots, B_K}$ with μ_1 in the diagonal elements. H is a square matrix of order $\left(m_{K-1}^{B_3, B_4, \dots, B_K} - m_{K-2}^{B_4, \dots, B_K}\right)$ with μ_1 in the diagonal elements.

Algorithm to Generate Matrix A

The following is a description of the algorithm to generate A , which was coded in C++. The user inputs K the number of stations, B_2, B_3, \dots, B_K the buffer capacities, the mean service rates $\mu_1, \mu_2, \dots, \mu_K$, and θ the feedback probability.

- RULE 1.**
- (i) The first element (row=1, column=1) is equal to $-\mu_1$ and element (row=1, column=2) equals μ_K .
 - (ii) This part generates the next $(B_K + 2)$ rows.
The next $(B_K + 2)$ diagonal elements (i, i) are put equal to $-\mu_1 - \mu_K$ and the value μ_K is placed in element $(i, i + 1)$, for the $(B_K + 2)$ rows, except the last row.

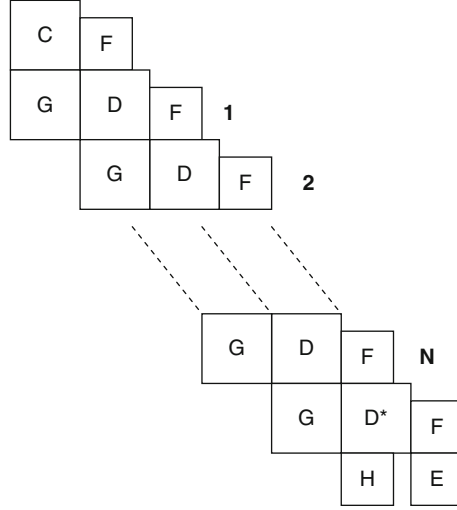


Fig. 2.5. Structure of A , $K > 2$, $B_2 = N$, B_3, B_4, \dots, B_K

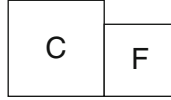


Fig. 2.6. Illustration of Rule 2

- (iii) If $K=2$, then μ_1 is added to the last diagonal element created above. This is matrix A for $K = 2$, go to Rule 6.

Rule 1 will create A if $K = 2$. If $K > 2$ it will create sub-matrix C for $K = 3$. Rules 2, 3, 4, and 5, below, are all contained within a loop (see ‘DO $T = 3$ to K ’ below). ‘END DO T ’ denotes the end of the loop. If $K = 3$, the first iteration of the loop will create A for $K = 3$, B_{K-1}, B_K , if not, then sub-matrix C for $K = 4$, B_{K-2}, B_{K-1}, B_K is created and so on until A for $K = K$ is created.

DO $T = 3$ to K

$$X = (T - 2)$$

$$Y = (X + 1)$$

$$W = (K - X)$$

RULE 2. Place the top left element of a square matrix of order $\left(m_Y^{B_{W+1}, \dots, B_K} - m_{Y-1}^{B_{W+2}, \dots, B_K}\right)$ with μ_W in its diagonal elements, in the $\left(m_{Y-1}^{B_{W+2}, \dots, B_K} + 1\right)$ row and the $\left(m_Y^{B_{W+1}, \dots, B_K} + 1\right)$ column of A .

This is sub-matrix F and its position relative to C is illustrated in Figure 2.6.

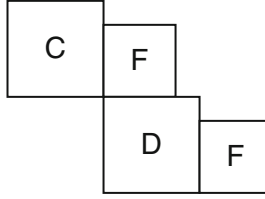


Fig. 2.7. Illustration of Rule 3

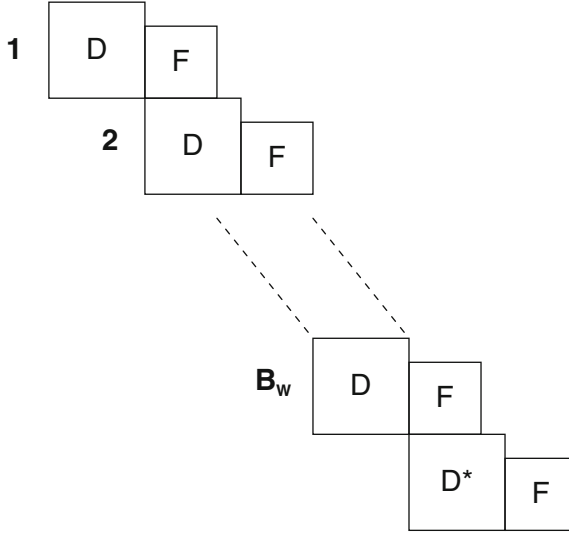


Fig. 2.8. Illustration of Rule 4

RULE 3. C is a square matrix of order $m_Y^{B_W+1, \dots, B_K}$. D is generated from C by subtracting μ_W from the first $\left(m_Y^{B_W+1, \dots, B_K} - m_{Y-1}^{B_W+2, \dots, B_K}\right)$ diagonal elements of C . D is positioned as in Figure 2.7. Also, F is copied onto F (see Figure 2.7). If $B = 0$ and $T = K$ then μ_1 is also added to the last $m_{Y-1}^{B_W+2, \dots, B_K}$ diagonal elements of C , i.e., C will be copied onto D^* .

Rule 4 is contained within a loop ('DO $Z = 1$ to B_W '), and it is executed B_W times. If $B_W = 0$, this rule is not used.

RULE 4. DO $Z = 1$ to B_W

Copy Sub-matrices D and F as described in Figure 2.8.

When $T = K$ and $Z = B_W$, μ_1 is added to

the last $m_{Y-1}^{B_W+2, \dots, B_K}$ diagonal elements of D ,

i.e., sub-matrix D is copied onto D^* .

END DO Z

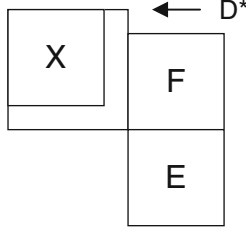


Fig. 2.9. Illustration of Rule 5

RULE 5. The top left of the square sub-matrix D^* of dimension $\left(m_Y^{B_{W+1}, \dots, B_K} - m_{Y-1}^{B_{W+2}, \dots, B_K}\right)$ is copied on to E , see Figure 2.9 i.e., X is copied onto E . The position of E is also illustrated in Figure 2.9. If $T = K$, μ_1 is added to all the diagonal elements of E .
END DO T

Rule 6 below generates the non-zero elements of A_2 and is executed after exiting ‘DO $T = 3$ to K ’.

RULE 6. DO $I = 1$ to $\left(m_K^{B_2, \dots, B_K} - m_{K-1}^{B_3, \dots, B_K}\right)$
Place μ_1 in row $\left(m_{K-1}^{B_3, \dots, B_K} + I\right)$ and column I .
END DO I

Application of the Algorithm

Here, the explicit derivation of the conservative matrix A for $K = 3, B_2 = 1, B_3 = 0$, with exponentially distributed processing times with mean values $\frac{1}{\mu_i}$, $i = 1, 2, 3$, is developed by applying the algorithm described above.

Rule 1

Applying Rule 1, matrix (2.31) is obtained. Since $K > 2$, this is matrix C for $K = 3$.

$$C = \begin{vmatrix} -\lambda - \mu_1 & \mu_3 & 0 \\ 0 & -\lambda - \mu_1 - \mu_3 & \mu_3 \\ 0 & 0 & -\lambda - \mu_1 - \mu_3 \end{vmatrix}. \quad (2.31)$$

Rule 1(i) generated the first row of C . Rule 1(ii) generated the next 2 = $(B_3 + 2)$ row(s): note that in the last row generated by Rule 1(ii), μ_3 is not placed in the column next to the diagonal.

Rules 2, 3, 4, 5 are all contained within a loop which is executed $(K - 3 + 1)$ times. Therefore, for the example illustrated here, only one iteration of the loop is performed, with $T = 3, X = 1, Y = 2$ and $W = 2$.

Rule 2

Rule 2 will generate a square matrix (sub-matrix F) of order $2 = (m_2^0 - m_1) = (m_Y^{B_W+1, \dots, B_K} - m_{Y-1}^{B_W+2, \dots, B_K})$ with $\mu_2 = \mu_W$ in the diagonal elements. The top left element of F is positioned in the $2^{nd} = (m_1 + 1) = (m_{Y-1}^{B_W+2, \dots, B_K} + 1)$ row and the $4^{th} = (m_2^0 + 1) = (m_Y^{B_W+1, \dots, B_K} + 1)$ column of A . The first 3 rows of matrix A are given in matrix (2.32).

$$\begin{vmatrix} -\mu_1 & \mu_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\mu_1 - \mu_3 & \mu_3 & \mu_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\mu_1 - \mu_3 & 0 & \mu_2 & 0 & 0 & 0 & 0 & 0 \end{vmatrix}. \quad (2.32)$$

Rule 3

Rule 3 generates sub-matrix D from sub-matrix C . D is generated from C by subtracting $\mu_2 = \mu_W$ from the first $2 = (m_2^0 - m_1) = (m_Y^{B_W+1, \dots, B_K} - m_{Y-1}^{B_W+2, \dots, B_K})$ diagonal elements of C . Rule 3 also copies sub-matrix F . The positions of sub-matrices D and F are illustrated in Figure 2.7. Excluding the non-diagonal μ_1 elements that are generated by Rule 6, matrix (2.33) gives the first 6 rows of A for $K = 3, B_2 = 1, B_3 = 0$

$$\begin{vmatrix} -\mu_1 & \mu_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\mu_1 - \mu_3 & \mu_3 & \mu_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\mu_1 - \mu_3 & 0 & \mu_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\mu_1 - \mu_2 & \mu_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\mu_1 - \mu_2 - \mu_3 & \mu_3 & \mu_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\mu_1 - \mu_3 & 0 & \mu_2 & 0 & 0 \end{vmatrix} \quad (2.33)$$

Rule 4

Rule 4 copies sub-matrix D $B_W = 1$ times. Because $T = K$ and $Z = B_W$, μ_1 is added to the last $m_1 = 1$ diagonal elements of D , i.e., sub-matrix D is copied onto D^* (see Figure 2.8). Matrix (2.34) gives the first nine rows of A (excluding the non-diagonal elements generated by Rule 6), which have been generated by rules 1, 2, 3 and 4.

$$\begin{vmatrix} -\mu_1 & \mu_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\mu_1 - \mu_3 & \mu_3 & \mu_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\mu_1 - \mu_3 & 0 & \mu_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\mu_1 - \mu_2 & \mu_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\sum_{i=1}^3 \mu_i & \mu_3 & \mu_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\mu_1 - \mu_3 & 0 & \mu_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\mu_1 - \mu_2 & \mu_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\sum_{i=1}^3 \mu_i & \mu_3 & \mu_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\mu_3 & 0 \end{vmatrix} \quad (2.34)$$

Rule 5

E is a square matrix of order $2 = (m_2^0 - m_1) = (m_Y^{B_W+1, \dots, B_K} - m_{Y-1}^{B_W+2, \dots, B_K})$ equal to the top left of sub-matrix D^* of the said dimension (see Figure 2.9). Since $T = K$, μ_1 is added to all the diagonal elements of E . The position of E is illustrated in Figure 2.9, the top left element of E is positioned on the 10^{th} row and the 10^{th} column of A (see matrix (2.35)).

Rule 6

The following loop generates the elements of A_2 (the non-diagonal μ_1 elements of A), $8 = (m_3^{1,0} - m_2^0) = (m_K^{B_2, \dots, B_K} - m_{K-1}^{B_3, \dots, B_K})$ and $3 = m_2^0 = m_{K-1}^{B_3, \dots, B_K}$.

DO $I = 1$ to 8

Place μ_1 in row $(3 + I)$ and column I .

END DO I

The required matrix A for $K = 3, B_2 = 1, B_3 = 0$ is given in matrix (2.35).

$$A = \begin{vmatrix} -\mu_1 & \mu_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\mu_1 - \mu_3 & \mu_3 & \mu_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\mu_1 - \mu_3 & 0 & \mu_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mu_1 & 0 & 0 & -\mu_1 - \mu_2 & \mu_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \mu_1 & 0 & 0 & -\sum_{i=1}^3 \mu_i & \mu_3 & \mu_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mu_1 & 0 & 0 & -\mu_1 - \mu_3 & 0 & \mu_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu_1 & 0 & 0 & -\mu_1 - \mu_2 & \mu_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & -\sum_{i=1}^3 \mu_i & \mu_3 & \mu_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & -\mu_3 & 0 & \mu_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & -\mu_2 & \mu_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_1 & 0 & 0 & -\mu_2 - \mu_3 \end{vmatrix} \quad (2.35)$$

2.1.3 A simple non-linear flow model

Non-linear flow models have the characteristic that parts may be returned to upstream stations or skip stations or meet other parts at particular stations for assembly or two or more parts emerge from a disassembly station. Thus, non-linearity implies some lack of strict successive continuity of a distinct product going from one station to the succeeding station in a production line.

In non-linear flow models consideration is given to assembly/disassembly and merge operations in production lines. These models may also take account of quality inspection stations and allow for the possibility of rework where a product is returned to earlier stations. Clearly, the topology of non-linear flow is more complicated than linear flow models.

Here, consideration is given to the non-linear flow model shown in Figure 2.10.

The merge phenomenon is indicated in Figure 2.10. Two machines upstream from the buffer perform the same operation and feed the buffer in such a way that one machine has priority over the other when the buffer is full. The third machine

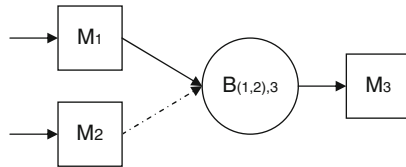


Fig. 2.10. A merge non-linear flow model

removes material from the buffer. The circle indicates a buffer of finite capacity and the squares indicate the machines. Thus the priority-one buffer is always selected first unless it is empty, where the priority-two buffer is chosen. The machines may break down.

Among the major assumptions of the model investigated here, are that the two upstream machines are never starved, the third machine is never blocked, all machines have equal and constant processing times, and geometrically distributed repair times and times to failures and machines can only fail while processing. The phenomenon of partial and full blocking of the second upstream machine is taken into account.

In order to analyze the model of Figure 2.10, the following three methodological steps are required:

- (i) Derivation of the transition equations of all states of the system (internal, lower boundary and upper boundary).
- (ii) Development of a recursive algorithm for generating the transition matrix for any value C of the extended storage level of buffer $B_{(1,2),3}$ (that is, the capacity of the original buffer plus 3).
- (iii) Numerical computation of the transition probabilities and then of the various performance measures of the system.

A formula for the number of states, m , for any value $C > 4$ of the extended storage level of buffer $B_{(1,2),3}$ is given by

$$m = (8 \times C) - 4. \quad (2.36)$$

The expected in-process inventory (average buffer level), \overline{WIP} , of the system of Figure 2.10 may be written as follows:

$$\overline{WIP} = \sum_{c=0}^C \sum_{\alpha_1=0}^1 \sum_{\alpha_2=0}^1 \sum_{\alpha_3=0}^1 c p[c, \alpha_1, \alpha_2, \alpha_3] \quad (2.37)$$

where $p[c, \alpha_1, \alpha_2, \alpha_3]$ denotes the steady-state probability of the system being in state $[c, \alpha_1, \alpha_2, \alpha_3]$. The level of the extended buffer is denoted by c . $\alpha_i, i = 1, 2, 3$, denotes the status of machine M_i , which may be up ($\alpha_i = 1$) or down ($\alpha_i = 0$).

The blocking probabilities of machines M_1 and M_2 , denoted by p_1^{bl} , p_2^{bl} and the starvation probability of machine M_3 , denoted by p_3^{st} are

$$p_1^{bl} = p[C-1, 1, 0, 0] + p[C-1, 1, 0, 1] + p[C, 1, 1, 0] + p[C, 1, 1, 1], \quad (2.38)$$

$$p_2^{bl} = p[C-1, 0, 1, 0] + p[C-1, 0, 1, 1] + p[C-1, 1, 1, 0] \\ + p[C-1, 1, 1, 1] + p[C, 1, 1, 0] + p[C, 1, 1, 1], \quad (2.39)$$

$$p_3^{st} = p[0, 0, 0, 1] + p[0, 0, 1, 1] + p[0, 1, 0, 1] + p[0, 1, 1, 1]. \quad (2.40)$$

The mean production rates related to each one of the three machines can be readily determined. If β_i and r_i are the mean rates of failure and repair, respectively, of machine M_i , then $e_i = r_i / (r_i + \beta_i)$, $i = 1, 2, 3$ represents the fraction of time that machine M_i is operational. Since all processing times are identical and are taken as the time unit, it is obvious that e_i , $i = 1, 2, 3$, is the isolated mean production rate of machine M_i , i.e., the mean production rate of machine M_i , if it were working alone. Since machines M_i , $i = 1, 2, 3$ are part of the system, blocking and starvation probabilities should be taken into account. Therefore the mean production rates (throughputs) related to each one of these three machines are

$$X_1 = (1 - p_1^{bl})e_1 \quad (2.41)$$

$$X_2 = (1 - p_2^{bl})e_2 \quad (2.42)$$

$$X_3 = (1 - p_3^{st})e_3. \quad (2.43)$$

In order to determine the throughput of the system shown in Figure 2.10, the throughput of the third machine should be computed. The throughput, X , of the system is simply given by X_3 .

$$X = X_3. \quad (2.44)$$

In Diamantidis, Papadopoulos and Vidalis (2004), a process for the generation of the transition matrix was developed and an algorithm to evaluate the performance parameters, including the average buffer level and the throughput of the system, was presented.

2.2 Decomposition Approach

Queueing networks are a natural way of analyzing production lines. Although there is a very rich literature in queueing networks, difficulties in analysis arise when finite buffers are considered because of the associated starving and blocking phenomena. Classically, queueing networks tended to be investigated through a process of decomposition into a set of single-server systems. Such an approach is totally valid in the case of infinite buffers. With finite buffers, exact classical decomposition is inappropriate. However, many researchers have developed efficient decomposition methods for the approximate evaluation of tandem queues which are suitable for the analysis of production lines. As noted above, numerical techniques based on exact Markovian analysis are space and computer time consuming for solutions to the exact queueing problems and are generally only applicable in practice to small production lines. Considerable effort has been expended on the development of approximate solutions to large-scale production lines.

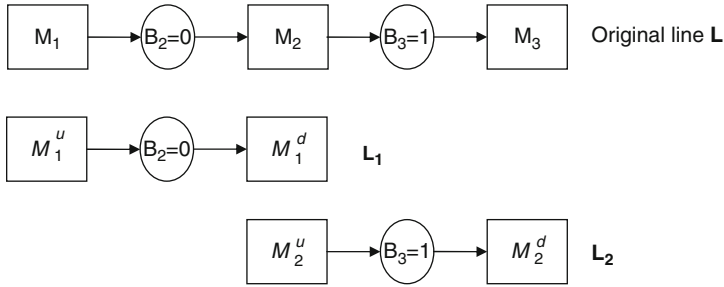


Fig. 2.11. A three-station line, L , decomposed into two sub-lines, L_1 and L_2

Essentially, the decomposition approach as applied to a K -station line consists of decomposing the original line into a set of $K - 1$ sub-lines. Each sub-line normally consists of two stations and an intermediate buffer which corresponds to a buffer of the original line. The original method proposed by Gershwin (1987) was initially used in the analysis of serial production lines. Consider the following example.

Example: Consider a balanced three-station production line where each workstation consists of only one machine (this is the original line L). All machines are assumed to be perfectly reliable with exponentially distributed service times and identical mean service rates, $\mu_1 = \mu_2 = \mu_3 = 1$. There is no intermediate buffer between the first two stations, whereas there is an intermediate buffer of size 1 between the second and third station, viz., $B_2 = 0$ and $B_3 = 1$. The original line, L , is decomposed into two sub-lines, L_1 and L_2 , as shown in Figure 2.11. X_K denotes the throughput of the K -station line, whereas X_1 and X_2 denote the throughput of sub-line L_1 and sub-line L_2 , respectively.

Following the approach of Gershwin (1994):

The *boundary conditions* of the decomposition method are

$$\mu_1^u = \mu_1, \quad (2.45)$$

$$\mu_{K-1}^d = \mu_K. \quad (2.46)$$

The general steps of the decomposition method are the following:

Step 1: Initialization

$$\mu_i^u = \mu_i, \quad i = 1, 2, \dots, K - 1$$

$$\mu_i^d = \mu_{i+1}, \quad i = 1, 2, \dots, K - 1.$$

Step 2: Iteration

Perform the following steps 2.1 and 2.2 alternately until the termination condition is satisfied.

Step 2.1: Evaluate quantities

$$\mu_i^u = \frac{1}{\frac{1}{X_{i-1}} + \frac{1}{\mu_i} - \frac{1}{\mu_{i-1}^d}}, \quad i = 2, 3, \dots, K - 1$$

Step 2.2: Evaluate quantities

$$\mu_i^d = \frac{1}{\frac{1}{X_{i+1}} + \frac{1}{\mu_{i+1}} - \frac{1}{\mu_{i+1}^u}}, \quad i = K-2, K-3, \dots, 1$$

Step 3: Termination condition

The algorithm is terminated when

$$|X_i - X_1| < \varepsilon, \quad i = 2, 3, \dots, K-1,$$

where ε is a pre-determined very small positive real number.

Application of the above decomposition algorithm to the example three-station balanced production line, L :

INITIALIZATION: $\mu_1^u = \mu_1$, $\mu_2^u = \mu_2$, $\mu_1^d = \mu_2$, $\mu_2^d = \mu_3$.

FIRST ITERATION (I1)

Step 2.1: From the boundary condition: $\mu_1^u = \mu_1 = 1$ and

$$\begin{aligned} \mu_2^u &= \frac{1}{\frac{1}{X_1} + \frac{1}{\mu_1} - \frac{1}{\mu_1^d}} \\ \mu_1^d &\stackrel{\mu_2}{=} \frac{1}{\frac{1}{X_1} + \frac{1}{1} - \frac{1}{\mu_2}} \\ &= \frac{1}{\frac{1}{0.666667} + \frac{1}{\mu_1} - \frac{1}{1}} = 0.666667, \end{aligned}$$

where X_1 is the throughput of sub-line L_1 which is calculated from the Markovian algorithm with the parameter values $\mu_1^u = \mu_1 = 1$, $\mu_1^d = \mu_2 = 1$ and $B_2 = 0$ and gives the value $X_1 = 0.666667$.

Step 2.2: From the boundary condition: $\mu_2^d = \mu_3 = 1$ and

$$\begin{aligned} \mu_1^d &= \frac{1}{\frac{1}{X_2} + \frac{1}{\mu_2} - \frac{1}{\mu_2^d}} \\ &\stackrel{\text{Step 2.1}}{=} \frac{1}{\frac{1}{X_2} + \frac{1}{1} - \frac{1}{0.666667}} \\ &= \frac{1}{\frac{1}{0.584573} + \frac{1}{\mu_1} - \frac{1}{0.666667}} = 0.826001, \end{aligned}$$

where X_2 is the throughput of sub-line L_2 which is calculated from the Markovian algorithm with the parameter values $\mu_2^u = 0.666667$, $\mu_2^d = \mu_3 = 1$ and $B_3 = 1$ and gives the value $X_2 = 0.584573$.

TERMINATION CONDITION: X_1 is the throughput of sub-line L_1 which is calculated from the Markovian algorithm with the parameter values $\mu_1^u = \mu_1 = 1$, $\mu_1^d = 0.826001$ and $B_2 = 0$ and gives the values $X_1 = 0.601320$ and $X_2 = 0.584573$

calculated in step 2.2 above. Thus, $|X_2 - X_1| > \varepsilon = 0.0001$ and the two-step iteration continues.

SECOND ITERATION (I2)

Step 2.1: From the boundary condition: $\mu_1^u = \mu_1 = 1$ and

$$\begin{aligned}\mu_2^u &= \frac{1}{\frac{1}{X_1} + \frac{1}{\mu_1} - \frac{1}{\mu_1^d}} \\ \text{Step 2.2(I1)} \quad &= \frac{1}{\frac{1}{X_1} + 1 - \frac{1}{0.826001}} \\ &= \frac{1}{\frac{1}{0.601320} + 1 - \frac{1}{0.826001}} = 0.688536.\end{aligned}$$

Step 2.2: From the boundary condition: $\mu_2^d = \mu_3 = 1$ and

$$\begin{aligned}\mu_1^d &= \frac{1}{\frac{1}{X_2} + \frac{1}{\mu_2} - \frac{1}{\mu_2^u}} \\ \text{Step 2.1(I2)} \quad &= \frac{1}{\frac{1}{X_2} + 1 - \frac{1}{0.688536}} \\ &= \frac{1}{\frac{1}{0.598217} + 1 - \frac{1}{0.688536}} = 0.820157,\end{aligned}$$

where X_2 is the throughput of sub-line L_2 which is calculated from the Markovian algorithm with the parameter values $\mu_2^u = 0.688536$, $\mu_2^d = \mu_3 = 1$ and $B_3 = 1$ and gives the value $X_2 = 0.598217$.

TERMINATION CONDITION: X_1 is the throughput of sub-line L_1 which is calculated from the Markovian algorithm with the parameter values $\mu_1^u = \mu_1 = 1$, $\mu_1^d = 0.820157$ and $B_2 = 0$ and gives the values $X_1 = 0.598823$ and $X_2 = 0.598217$ calculated in step 2.2 above. Again $|X_2 - X_1| > \varepsilon = 0.0001$ and the two-step iteration continues.

THIRD ITERATION (I3)

Step 2.1: From the boundary condition: $\mu_1^u = \mu_1 = 1$ and

$$\begin{aligned}\mu_2^u &= \frac{1}{\frac{1}{X_1} + \frac{1}{\mu_1} - \frac{1}{\mu_1^d}} \\ \text{Step 2.2(I2)} \quad &= \frac{1}{\frac{1}{X_1} + 1 - \frac{1}{0.820157}} \\ &= \frac{1}{\frac{1}{0.598823} + 1 - \frac{1}{0.820157}} = 0.689339.\end{aligned}$$

Step 2.2: From the boundary condition: $\mu_2^d = \mu_3 = 1$ and

$$\begin{aligned}\mu_1^d &= \frac{1}{\frac{1}{X_2} + \frac{1}{\mu_2} - \frac{1}{\mu_2^u}} \\ &\stackrel{\text{Step 2.1 (I3)}}{=} \frac{1}{\frac{1}{X_2} + \frac{1}{1} - \frac{1}{0.689339}} \\ &= \frac{1}{\frac{1}{0.598707} + 1 - \frac{1}{0.689339}} = 0.819940,\end{aligned}$$

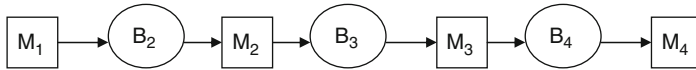
where X_2 is the throughput of sub-line L_2 which is calculated from the Markovian algorithm with the parameter values $\mu_2^u = 0.689339$, $\mu_2^d = \mu_3 = 1$ and $B_3 = 1$ and gives the value $X_2 = 0.598707$.

TERMINATION CONDITION: X_1 is the throughput of sub-line L_1 which is calculated from the Markovian algorithm with the parameter values $\mu_1^u = \mu_1 = 1$, $\mu_1^d = 0.819940$ and $B_2 = 0$ and gives the values $X_1 = 0.598738$ and $X_2 = 0.598707$ calculated in step 2.2 above. Now, $|X_2 - X_1| = 0.000031 < \varepsilon = 0.0001$ and the algorithm terminates giving a throughput value, $X_{\text{DECO}} = 0.5987$. This value may be compared against the throughput value calculated from the Markovian algorithm, $X_{\text{MARK}} = 0.613333$. The reader may note that if the value of ε was smaller, say $\varepsilon = 0.00001$, then more iterations would be needed for the algorithm to terminate.

A coded version of the original Gershwin's decomposition algorithm is not at the website associated with this book. However, the algorithm developed by Dr. Diamantidis (Diamantidis, Papadopoulos and Heavey, 2006) for parallel perfectly reliable machine production lines (given in Section 2.5 and at the website associated with this text with abbreviated name DECO-2) may be used by setting the number of parallel machines at each station equal to 1 as an alternative to the Gershwin algorithm for perfectly reliable single-machine stations. The authors have checked that the equations derived from Diamantidis et al. (2006) work, and setting the number of servers at each station equal to 1 showed them to be identical to those developed by Gershwin for the single-machine perfectly reliable production lines.

Dallery and his group undertook considerable work in the area of decomposition modeling. Available at the website associated with this book with abbreviated name DECO-1 is a coded version of an algorithm given in Dallery and Frein (1993) for the analysis of reliable production lines with single machines at each station. To illustrate the development of Dallery and Frein's algorithm, consider the four-station production line with finite inter-station buffers, B_2, B_3 and B_4 , with capacities B_2, B_3 and B_4 , respectively, shown in Figure 2.12. For simplicity, work-stations are denoted by M_i , $i = 1, 2, 3, 4$ instead of WS_i , $i = 1, \dots, 4$. Figure 2.13 depicts the decomposition of this four-station line into three sub-lines, L_1, L_2 , and L_3 , each consisting of two stations and one intermediate buffer. All single-machine work-stations are assumed to be perfectly reliable and the service times at each machine are exponentially distributed with mean service rates, μ_i , $i = 1, \dots, 4$.

The application of queueing network theory to production lines involved the use of either the open model or the saturated model. In the saturated model, the assumption is that the first station is never starved, whereas in the open model, the first queue



M_i : work-station i , $i = 1, 2, 3, 4$

B_i : buffer i , $i = 2, 3, 4$

Fig. 2.12. Production line, L , with $K = 4$ work-stations and 3 intermediate buffers

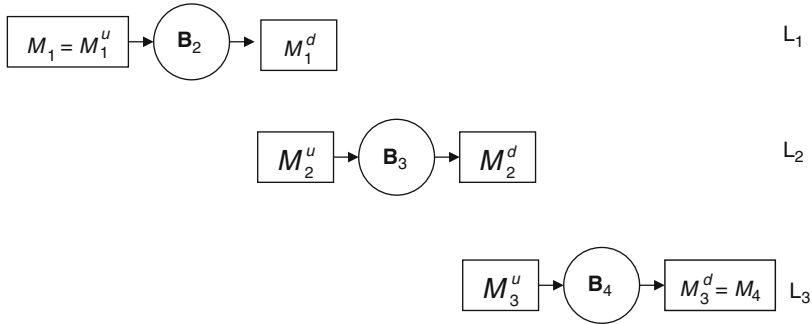


Fig. 2.13. Decomposition of the original line, L , into three sub-lines each with two stations and one buffer

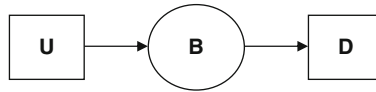


Fig. 2.14. Sub-line L_i

is finite and the first station may be starved. The model considered here is a saturated model. Actually, all models examined in this book are saturated models except for the model solved via the expansion method (Section 2.3).

Some formal results from queueing theory are required in the development of the decomposition equations and these are given immediately below.

Queueing theory analysis of the sub-lines

Consider the sub-line shown in Figure 2.14.

There are $B + 2$ states in the system of Figure 2.14, where B is the buffer size of the intermediate buffer B . Some of the performance parameters of a two-station system, called decomposition block, are required for the development of the decomposition equations. The states of the system are characterized by $v = 0, 1, \dots, B + 1$, where v denotes the number of jobs waiting for service at station 2. $v = B + 1$ occurs when station U is blocked. μ_U and μ_D , respectively, denote the mean service rate of machines U and D . Let $p(v)$ be the steady-state probability of the system being in

state v and let $p_U(v)$ be the probability that there are v jobs in the buffer when a job is completed at station U . Likewise, let $p_D(v)$ be the probability that on completion of service at station D , there are v jobs left in buffer B . The throughput of the system is denoted by X . Using well-known results, the following relationships apply:

$$p_U(v) = \frac{p(v)}{1 - p(B+1)}, \quad v = 0, \dots, B \quad (2.47)$$

$$p_D(v) = \frac{p(v+1)}{1 - p(0)}, \quad v = 0, \dots, B \quad (2.48)$$

$$X = \mu_U(1 - p(B+1)) \quad (2.49)$$

$$X = \mu_D(1 - p(0)). \quad (2.50)$$

Two further probabilities are of interest, these being p_U^{bl} , the probability that station U is blocked, and p_D^{st} , the probability that station D is starved. Clearly these are given by

$$p_U^{bl} = p_U(B) \quad (2.51)$$

$$p_D^{st} = p_D(0). \quad (2.52)$$

Returning to the decomposition process, in Figure 2.13, sub-line L_i , $i = 1, 2, 3$ ($= K - 1$) approximates the flow of jobs in buffer B_i , $i = 2, 3, 4$ ($= K$) of the original line. In sub-lines L_i , $i = 1, \dots, K - 1$, stations M_i^u , $i = 1, \dots, K - 1$ and M_i^d , $i = 2, \dots, K - 1$ represent the part of the original line, L , upstream and downstream of buffer B_{i+1} , $i = 1, \dots, K - 1$, respectively. The concept is that whereas the buffer B_i is in all respects identical to the buffer preceding station i in the original line L , the two stations M_i^u and M_i^d are not identical to stations $i - 1$ and i , except that station M_1^u is identical to station M_1 and station M_3^d is identical to station M_4 . The characteristics of the remaining stations are so chosen that in effect they represent the impact of the upstream and downstream parts of the production line L on the buffer B_i , $i = 2, 3, 4$. μ_i , $i = 1, \dots, K$ denotes the mean service rate of station i , $i = 1, \dots, K$ in the original line, L . Similarly, μ_i^u , $i = 1, \dots, K - 1$ and μ_i^d , $i = 1, \dots, K - 1$ denote the mean service rate of stations M_i^u , $i = 1, \dots, K - 1$ and M_i^d , $i = 1, \dots, K - 1$, respectively, in the sub-lines, L_i , $i = 1, \dots, K - 1$. All service times are assumed to be exponentially distributed with the respective mean service rates given above.

The development of the sets of the decomposition equations starts out by considering the sub-lines L_1 and L_3 . This gives the boundary conditions already stated above that $\mu_1^u = \mu_1$ and $\mu_3^d = \mu_4 = \mu_K$.

In total, there are in general $2(K - 1)$ unknowns, and there is a need to obtain another $2(K - 2)$ independent equations.

$w_i = 1/\mu_i$, $i = 1, \dots, K$ denotes the mean service time (average work-load) at station i , $i = 1, \dots, K$ of the original line, L . The mean service time of the downstream station M_i^d , $i = 1, 2, \dots, K - 2$ ($K - 2 = 4 - 2 = 2$ for the example line of Figure 2.12 and the sub-lines of Figure 2.13), denoted by $w_i^d = 1/\mu_i^d$, is the sum of the service time at station i in the original line and the possible blocking time of station M_i in the

original line L , which is equivalent to the blocking time of station M_{i+1}^u in sub-line L_{i+1} due to the fact that buffer B_{i+2} is full and on the assumption that the station is perfectly reliable. This gives rise, in general, to the following set of equations for the reliable exponential production lines:

$$w_i^d = w_i + p_{i+1}^{bl} w_{i+1}^d, \quad i = 1, 2, \dots, K-2, \quad (2.53)$$

where, p_{i+1}^{bl} denotes the blocking probability of station M_{i+1}^u .

A similar set of equations may be developed for the upstream stations. More specifically, the mean service time of the upstream station $M_i^u, i = 2, \dots, K-1$ ($K-1 = 4-1 = 3$ for the example line of Figure 2.12 and the sub-lines of Figure 2.13), denoted by $w_i^u = 1/\mu_i^u$, is the sum of the service time of station $i-1$ in the original line and the possible starvation time of station $i-1$. The latter event in the original line is equivalent to the starvation of station M_{i-1}^d in sub-line L_{i-1} . This gives rise, in general, to the following set of equations for the reliable exponential production lines:

$$w_i^u = w_{i-1} + p_{i-1}^{st} w_{i-1}^u, \quad i = 2, 3, \dots, K-1, \quad (2.54)$$

where p_{i-1}^{st} denotes the starvation probability of station M_{i-1}^d .

The third set of equations is related to the conservation of flow, i.e., the throughput of all stations in the line is the same and consequently the throughput of the sub-lines must satisfy the following flow equations:

$$X_1 = X_2 = \dots = X_{K-1}, \quad (2.55)$$

where X_i denotes the throughput of sub-line $L_i, i = 1, \dots, K-1$.

As may be noted from the above, there are two sets of $K-2$ equations plus two boundary conditions, so it is not necessary to use all the equations to solve for the unknowns. This leads to the utilization of the following sub-set of the above equations:

$$w_i^u = w_{i-1} + p_{i-1}^{st} w_{i-1}^u, \quad i = 2, 3, \dots, K-1, \quad (2.56)$$

$$w_i^d = w_i + p_{i+1}^{bl} w_{i+1}^d, \quad i = 1, 2, \dots, K-2, \quad (2.57)$$

$$w_1^u = w_1, \quad \text{and} \quad w_{K-1}^d = w_K \quad (2.58)$$

Dallery and Frein (1993) proved that the above set of equations satisfies the conservation of flow criterion. They also proved the existence and uniqueness of the solution derived from this set of equations and that this symmetrical set of equations is equivalent to each of the following sets of equations:

$$w_i^d = w_i + p_{i+1}^{bl} w_{i+1}^d, \quad i = 1, 2, \dots, K-2, \quad (2.59)$$

$$X_1 = X_2 = \dots = X_{K-1}, \quad (2.60)$$

$$w_1^u = w_1, \quad \text{and} \quad w_{K-1}^d = w_K. \quad (2.61)$$

$$w_i^u = w_{i-1} + p_{i-1}^{st} w_{i-1}^u, \quad i = 2, 3, \dots, K-1, \quad (2.62)$$

$$X_1 = X_2 = \dots = X_{K-1}, \quad (2.63)$$

$$w_1^u = w_1, \quad \text{and} \quad w_{K-1}^d = w_K. \quad (2.64)$$

Iterative procedures for solving the above three sets of equations have been proposed by Dallery and Frein (1993) and form the basis of the decomposition algorithm available at the website associated with this book.

The numerical processes involved in the algorithm are relatively straightforward. The two boundary conditions on the mean service rates of the first and last stations of the line are set and then the mean service rate of each of the other downstream stations are set equal to the values of the original line. The starvation and blocking probabilities are then calculated and values of the upstream and changed values of the downstream stations mean service rates are developed. This process continues until satisfactory convergence is achieved. Finally, the throughput of the line may be determined. The numerical decomposition process is outlined in flow diagram form in Figure 2.15.

In general, the decomposition method as applied to production lines consists essentially of three steps as follows:

1. The specification of the sub-lines
2. The determination of a set of equations used to evaluate the unknown parameters of each sub-line in such a way that the flow of material through the sub-lines resembles the corresponding flow of material in the original line. More specifically, the following conditions have to be satisfied as explicitly given in Gershwin (1994):
 - The rate of flow into and out of buffer B_i in sub-line L_i approximates that of buffer B_i in the original line L .
 - The probability of the buffer of sub-line L_i being empty or full is close to that of B_i in the original line L being empty or full.
 - The probability of resumption of flow into and out of the buffer in sub-line L_i in a time interval after a period during which it was interrupted is close to the probability of the corresponding event in the original line L .
 - The average level of material in buffer B_i in sub-line L_i approximates the corresponding material level in buffer B_i in the original line L .
3. The development of an appropriate procedure for solving the set of equations.

2.3 The Expansion Method

The expansion method is an approximation technique developed by Kerbache (1984), published also in Kerbache and MacGregor Smith (1987) and extended by Jain and MacGregor Smith (1994). This method is characterized as a combination of repeated trials and node-by-node decomposition solution procedures. Methodologies for computing performance measures for a finite queuing network use the following two kinds of blocking:

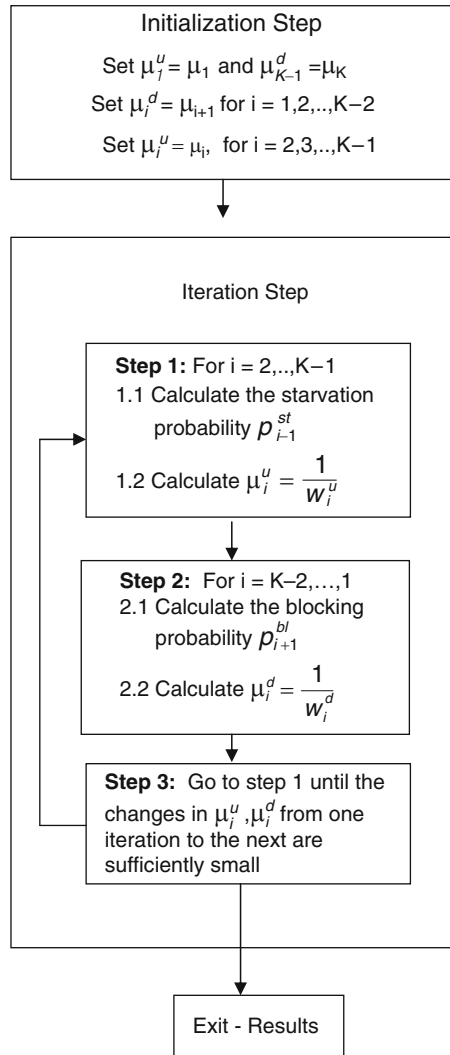


Fig. 2.15. Flow chart for decomposition method

- Type I:** The upstream node i gets blocked if the service on a unit is completed but it cannot move downstream due to the queue at the downstream node j being full. This is referred to as blocking after service (BAS) (Onvural and Perros, 1986, Perros, 1994).
- Type II:** The upstream node is blocked when the downstream node becomes saturated and service must be suspended on the upstream unit regardless of whether service is completed or not. This is referred to as blocking before service (BBS) (Onvural and Perros, 1986, Perros, 1994).

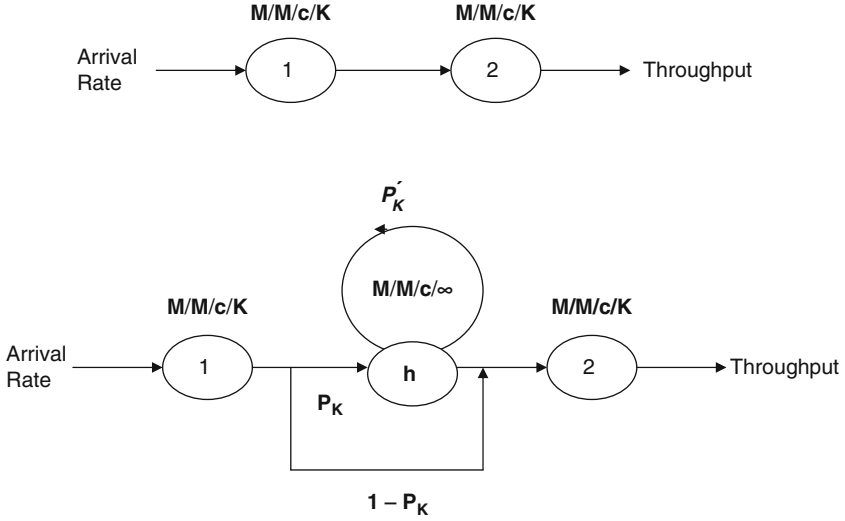


Fig. 2.16. Expansion of a finite queue $M/M/c/K$

The Expansion Method uses *Type I* blocking, which is prevalent in most production, manufacturing, transportation and other similar systems.

Consider a single node with finite capacity K (including service). This node essentially oscillates between two states—the saturated phase and the unsaturated phase. In the unsaturated phase, node j has at most $K - 1$ units (in service or in the queue). On the other hand, when the node is saturated no more units can join the queue. Refer to Figure 2.16 for a graphical representation of the expansion of a finite queue $M/M/c/K$. The reader may note that this model is the only open model considered in this book. All the other models are saturated models.

The Expansion Method consists of the following three stages:

- Stage I: Network reconfiguration.
- Stage II: Parameter estimation.
- Stage III: Feedback elimination.

The following notation defined by Kerbache and MacGregor Smith (1987) and Jain and MacGregor Smith (1994) will be used in further discussion regarding this methodology:

h := The holding node established in the expansion method.

Λ := External Poisson arrival rate to the network.

λ_j := Poisson arrival rate to node j .

$\tilde{\lambda}_j$:= Effective arrival rate to node j .

μ_j := Exponential mean service rate at node j .

$\tilde{\mu}_j$:= Effective service rate at node j due to blocking.

p_K := Blocking probability of finite queue of size K .

p'_K := Feedback blocking probability in the expansion method.

p_0^j := Unconditional probability that there is no unit in the service channel at node j (either being served or being held after service).
 X := Mean production rate (throughput).

Stage I: Network reconfiguration

Using the concept of two phases at node j , an artificial node h is added for each finite node in the network to register blocked units. Figure 2.16 shows the additional delay, caused to units trying to join the queue at node j when it is full, with probability p_K . The units successfully join queue j with a probability $(1 - p_K)$. Introduction of an artificial node also dictates the addition of new arcs with p_K and $(1 - p_K)$ as the routing probabilities.

The blocked unit proceeds to the finite queue with probability $(1 - p'_K)$ once again after incurring a delay at the artificial node. If the queue is still full, it is re-routed with probability p'_K to the artificial node where it incurs another delay. This process continues until it finds a space in the finite queue. A feedback arc is used to model the repeated delays. The artificial node is modeled as an $M/M/\infty$ queue. The infinite number of servers is used simply to serve the blocked unit a delay time without queuing.

Stage II: Parameter estimation

This stage essentially estimates the parameters p_K , p'_K and μ_h utilizing known results for the $M/M/c/K$ model.

- p_K : Analytical results from the $M/M/c/K$ model provide the following expression for p_K :

$$p_K = \frac{1}{c^{K-c}c!} \left(\frac{\lambda}{\mu}\right)^K p_0 \quad (2.65)$$

where for $(\lambda/c\mu \neq 1)$

$$p_0 = \left[\sum_{n=0}^{c-1} \frac{1}{n!} \left(\frac{\lambda}{\mu}\right)^n + \frac{(\lambda/\mu)^c}{c!} \frac{[1 - (\lambda/c\mu)^{K-c+1}]}{(1 - \lambda/c\mu)} \right]^{-1} \quad (2.66)$$

and for $(\lambda/c\mu = 1)$,

$$p_0 = \left[\sum_{n=0}^{c-1} \frac{1}{n!} \left(\frac{\lambda}{\mu}\right)^n + \frac{(\lambda/\mu)^c}{c!} (K - c + 1) \right]^{-1}. \quad (2.67)$$

- p'_K : Since there is no closed form solution for this quantity, an approximation obtained by Labetoulle and Pujolle (1980), using diffusion techniques, is used:

$$p'_K = \left[\frac{\mu_j + \mu_h}{\mu_h} - \frac{\lambda [(x_2^K - x_1^K) - (x_2^{K-1} - x_1^{K-1})]}{\mu_h [(x_2^{K+1} - x_1^{K+1}) - (x_2^K - x_1^K)]} \right]^{-1} \quad (2.68)$$

where x_1 and x_2 are the roots to the polynomial:

$$\lambda - (\lambda + \mu_h + \mu_j)x + \mu_h x^2 = 0 \quad (2.69)$$

where $\lambda = \lambda_j - \lambda_h(1 - p'_K)$ and λ_j and λ_h are the actual arrival rates to the finite and artificial holding nodes respectively.

In fact, λ_j the arrival rate to the finite node is given by:

$$\lambda_j = \tilde{\lambda}_i(1 - p_K) = \tilde{\lambda}_i - \lambda_h. \quad (2.70)$$

If an arriving unit is blocked, the queue is full and thus a unit is being serviced, so the arriving unit to the holding node has to remain in service at the artificial holding node for the remaining service time interval of the unit in service. The delay distribution of a blocked unit at the holding node has the same distribution as the remaining service time of the unit being serviced at the node doing the blocking. Using renewal theory, one can show that the remaining service time distribution has the following rate μ_h :

$$\mu_h = \frac{2\mu_j}{1 + \sigma^2\mu_j^2} \quad (2.71)$$

where σ^2 is the service time variance given by Kleinrock (1975). Notice that if the service time distribution at the finite queue doing the blocking is exponential with rate μ_j , then:

$$\mu_h = \mu_j$$

the service time at the artificial node is also exponentially distributed with rate μ_j .

Stage III: Feedback elimination

Due to the feedback loop around the holding node, there are strong dependencies in the arrival processes. Elimination of these dependencies requires reconfiguration of the holding node which is accomplished by recomputing the service time at the node and removing the feedback arc. The new service rate is given by:

$$\mu'_h = (1 - p'_K)\mu_h. \quad (2.72)$$

The probabilities of being in any of the two phases (saturated or unsaturated) are p_K and $(1 - p_K)$, respectively. The mean service time at a node i , preceding the finite node is μ_i^{-1} if in the unsaturated phase and $(\mu_i^{-1} + \mu_h'^{-1})$ in the saturated phase. Thus, on average, the mean service time at the node i preceding a finite node, is given by:

$$\tilde{\mu}_i^{-1} = \mu_i^{-1} + p_K\mu_h'^{-1}. \quad (2.73)$$

Similar equations can be established with respect to each of the finite nodes. Ultimately, a set of simultaneous non-linear equations in variables p_K , p'_K , μ_h^{-1} along with auxiliary variables such as μ_j and $\tilde{\lambda}_i$ is developed. Solving these equations

simultaneously, all the performance measures of the network can be computed:

$$\lambda = \lambda_j - \lambda_h(1 - p'_K) \quad (2.74)$$

$$\lambda_j = \tilde{\lambda}_i(1 - p_K) \quad (2.75)$$

$$\lambda_j = \tilde{\lambda}_i - \lambda_h \quad (2.76)$$

$$p'_K = \left[\frac{\mu_j + \mu_h}{\mu_h} - \frac{\lambda[(x_2^K - x_1^K) - (x_2^{K-1} - x_1^{K-1})]}{\mu_h[(x_2^{K+1} - x_1^{K+1}) - (x_2^K - x_1^K)]} \right]^{-1} \quad (2.77)$$

$$z = (\lambda + 2\mu_h)^2 - 4\lambda\mu_h \quad (2.78)$$

$$x_1 = \frac{[(\lambda + 2\mu_h) - z^{\frac{1}{2}}]}{2\mu_h} \quad (2.79)$$

$$x_2 = \frac{[(\lambda + 2\mu_h) + z^{\frac{1}{2}}]}{2\mu_h} \quad (2.80)$$

$$p_K = \frac{1}{c^{K-c}c!} \left(\frac{\lambda}{\mu} \right)^K p_0. \quad (2.81)$$

Equations (2.74) to (2.77) are related to the arrivals and feedback in the *holding* node. Equations (2.78) to (2.80) are used for solving equation (2.77) with z used as a dummy parameter for simplicity of the solution. Finally, equation (2.81) gives the approximation to the blocking probability derived from the exact model for the $M/M/c/K$ queue. Hence, essentially there are five equations to solve, viz. (2.74) to (2.77) and (2.81).

To recapitulate, first the network is expanded with an artificial holding node; this stage is followed by the approximation of the routing probabilities, due to blocking, and the service delay in the holding node; and, finally, the feedback arc at the holding node is eliminated. Once these three stages are completed, an expanded network has been developed which can then be used to compute the performance measures for the original network. As a decomposition technique, this approach allows the successive addition of a holding node for every finite node, estimation of the parameters and subsequent elimination of the holding node.

The expansion algorithm is available on the website associated with this text with the abbreviated name EXPAN. Not many practitioners are aware of the expansion method and there is little guidance in the published literature as to the accuracy achieved using the method in the analysis of realistic systems of interest to the designers of production lines. However, it must be recognized that in a historical context, the expansion method was used as a first serious attempt to computationally solve systems with parallel machines at each station.

2.4 The Aggregation Method

Lim, Meerkov and Top (1990) published an approximation approach used in the analysis of transfer lines which has come to be known as the aggregation method. This very powerful method begins by combining the first two machines of the transfer

line into a new combined machine. This aggregated combined machine is then combined with the third machine and this forward aggregation process is continued until the last machine is reached. A backward aggregation process is then applied. The algorithm which is available at the website associated with this book stops when the results of both aggregations (forward and backward) coincide.

Assumptions of the model

A serial transfer line is considered consisting of K machines and $K - 1$ intermediate buffers. Machines are assumed to have identical cycle time and the time axis is slotted with the slot/period duration equal to the cycle time. It is assumed that the first machine is never starved and the last machine is never blocked. It is further assumed that a certain machine i , $i = 1, \dots, K$ produces a part during any time slot/period with probability q_i and fails to do so with probability $1 - q_i$, provided that machine i is neither blocked nor starved. Mathematically, q_i is defined as follows:

$$q_i = 1 - \varepsilon \Lambda_i,$$

where $0 < \varepsilon \ll 1$, which characterizes the asymptotically reliable line, and Λ_i , $i = 1, \dots, K$ is independent of ε . The Λ_i 's were defined by Lim, Meerkov and Top (1990) as the *loss parameters*. Lim et al. (1990) also defined the following function:

$$Q(\alpha, \nu) = \frac{1 - \alpha}{1 - \alpha^\nu}, \quad \alpha \in \mathbb{R}^+, \quad \nu \in [1, \infty). \quad (2.82)$$

The two-machine, one-buffer transfer line

A two-machine, one-buffer transfer line in steady state is equivalent to a single aggregated machine characterized by

$$q_{\text{aggregation}} = 1 - \left[\Lambda_2 + \Lambda_1 Q\left(\frac{\Lambda_2}{\Lambda_1}, \nu\right) \right] \varepsilon \quad (2.83)$$

Thus, the loss parameter of the equivalent aggregated machine is

$$\Lambda_{\text{aggregation}} = \Lambda_2 + \Lambda_1 Q\left(\frac{\Lambda_2}{\Lambda_1}, \nu\right), \quad (2.84)$$

where Λ_1 and Λ_2 are, respectively, the loss parameters of the first and second machine. The mean production rate, X_2 , of the two-machine, one-buffer system is given by

$$\begin{aligned} X_2 &= 1 - \left[\Lambda_2 + \Lambda_1 Q\left(\frac{\Lambda_2}{\Lambda_1}, \nu\right) \right] \varepsilon + O(\varepsilon^2) \\ &= 1 - \left[\Lambda_1 + \Lambda_2 Q\left(\frac{\Lambda_1}{\Lambda_2}, \nu\right) \right] \varepsilon + O(\varepsilon^2). \end{aligned} \quad (2.85)$$

It is obvious that

$$\Lambda_{\text{aggregation}} = \Lambda_1 + \Lambda_2 Q\left(\frac{\Lambda_1}{\Lambda_2}, v\right). \quad (2.86)$$

Equations (2.84) and (2.86) show that

$$\Lambda_{\text{aggregation}} = \Lambda_1 + \Lambda_2 Q\left(\frac{\Lambda_1}{\Lambda_2}, v\right) = \Lambda_2 + \Lambda_1 Q\left(\frac{\Lambda_2}{\Lambda_1}, v\right).$$

Longer lines

The above process can be generalized to the case of homogeneous asymptotically reliable serial transfer lines consisting of K machines and $K - 1$ intermediate buffers. The first two machines are combined into an aggregated machine with the loss parameter, Λ_2^f , defined by (2.84), i.e.,

$$\Lambda_2^f = \Lambda_2 + \Lambda_1 Q\left(\frac{\Lambda_2}{\Lambda_1}, v_1\right).$$

Superscript ‘f’ indicates that during the aggregation, one moves forward (from the first to the last machine). The aggregated machine, characterized by Λ_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:

$$\Lambda_3^f = \Lambda_3 + \Lambda_2^f Q\left(\frac{\Lambda_3}{\Lambda_2^f}, v_2\right).$$

At the i th step of this multi-stage aggregation process, one may obtain:

$$\Lambda_i^f = \Lambda_i + \Lambda_{i-1}^f Q\left(\frac{\Lambda_i}{\Lambda_{i-1}^f}, v_{i-1}\right) \quad (2.87)$$

and at the final step:

$$\Lambda_K^f = \Lambda_K + \Lambda_{K-1}^f Q\left(\frac{\Lambda_K}{\Lambda_{K-1}^f}, v_{K-1}\right).$$

The estimate of the mean production rate (throughput) obtained as a result of this aggregation is:

$$X_K^f = 1 - \left[\Lambda_K + \Lambda_{K-1}^f Q\left(\frac{\Lambda_K}{\Lambda_{K-1}^f}, v_{K-1}\right) \right] \varepsilon. \quad (2.88)$$

Because there is no proof that X_K^f is close to the real throughput of the production line with K machines in series and $K - 1$ intermediate buffers, another set of equations should be supplemented, but this time directed backwards instead of forwards. This

scheme is called backward aggregation and aggregates the line moving from the last machine to the first machine. The respective loss paramaters are:

$$\begin{aligned}\Lambda_{K-1}^b &= \Lambda_{K-1} + \Lambda_K Q\left(\frac{\Lambda_{K-1}^f}{\Lambda_K}, v_{K-1}\right) \\ \Lambda_{K-2}^b &= \Lambda_{K-2} + \Lambda_{K-1}^b Q\left(\frac{\Lambda_{K-2}^f}{\Lambda_{K-1}^b}, v_{K-2}\right) \\ \Lambda_j^b &= \Lambda_j + \Lambda_{j+1}^b Q\left(\frac{\Lambda_j^f}{\Lambda_{j+1}^b}, v_j\right) \\ \Lambda_1^b &= \Lambda_1 + \Lambda_2^b Q\left(\frac{\Lambda_1^f}{\Lambda_2^b}, v_1\right).\end{aligned}$$

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

$$\begin{aligned}\Lambda_i^f(s+1) &= \Lambda_i + \Lambda_{i-1}^f(s+1) Q\left(\frac{\Lambda_i^b(s)}{\Lambda_{i-1}^f(s+1)}, v_{i-1}\right), \quad i = 2, \dots, K \\ s &= 0, 1, \dots, \quad \Lambda_i^b(0) = \Lambda_i, \quad \Lambda_1^f(s) = \Lambda_1, \quad \Lambda_K^b(s) = \Lambda_K, \quad \forall s. \quad (2.89) \\ \Lambda_j^b(s+1) &= \Lambda_j + \Lambda_{j+1}^b(s+1) Q\left(\frac{\Lambda_j^f(s+1)}{\Lambda_{j+1}^b(s+1)}, v_j\right), \quad j = 1, \dots, K-1.\end{aligned}$$

Procedure (2.89) generates the following two sequences of throughput estimates:

$$\begin{aligned}X_K^f(s) &= 1 - \Lambda_K^f(s) \varepsilon \\ X_K^b(s) &= 1 - \Lambda_1^b(s) \varepsilon.\end{aligned} \quad (2.90)$$

The properties of these sequences are described in Lim, Meerkov and Top (1990). The aggregation algorithm is available at the website associated with this text with abbreviated name AGGRE.

2.5 Modeling of Production Lines with Parallel Reliable Machines at Each Station

The throughput of production lines may be increased by adding extra machines at stations. It should be understood that all machines at the stations are used provided there is work available. Here, attention is confined to lines with reliable machines and with exponential processing times. A K -work-station line with $S_i, i = 1, 2, \dots, K$ parallel machines at work-station i , denoted by WS_i , and with intermediate buffers $B_j, j = 2, \dots, K$ of capacities B_j is depicted in Figure 2.17.

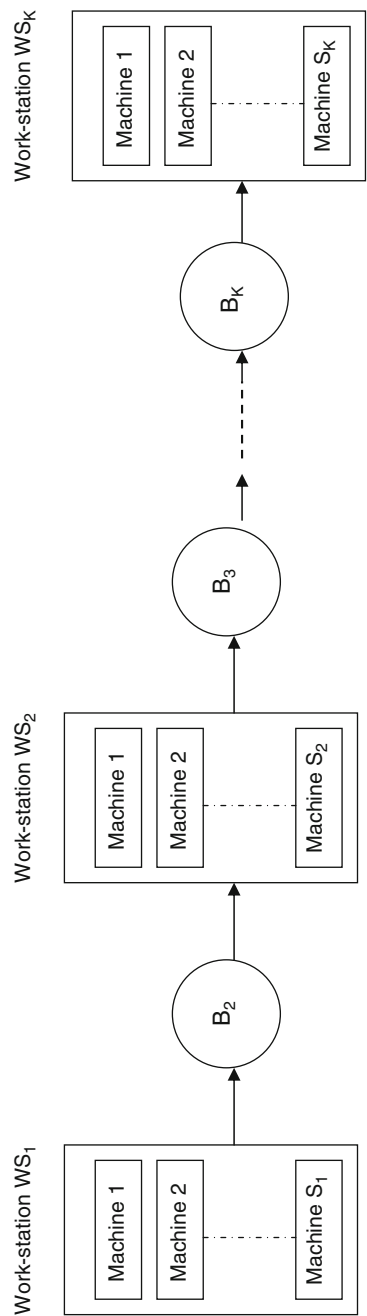


Fig. 2.17. A K -work-station production line with S_i parallel machines at each work-station $WS_i, i = 1, 2, \dots, K$

In sub-section 2.5.1, the exact solution to a two-station line with multiple machines at each station is presented. It might be noted that it is possible to develop the conservative matrix A of these systems with a view of developing exact numerical solutions along the lines already explained in Section 2.1. Interested readers might refer to Vidalis and Papadopoulos (2001). However, computational complexities considerably reduce the value of developing the conservative matrix A of such systems. In sub-section 2.5.2, an alternative exact solution to the two-station line with parallel machines at each station is presented as given in Diamantidis, Papadopoulos and Heavey (2006). This solution is used as a building block for a decomposition analysis of larger production lines with parallel machines at each station. Details of the latter analysis are given in sub-section 2.5.3.

2.5.1 Exact solution to a two-station production line with parallel machines at each station

Consider a system consisting of two stations with S_1 and S_2 parallel machines at station 1 and station 2, respectively. It is assumed that the first station is always busy, i.e., it is saturated and the intermediate buffer is of capacity B_2 which includes the number of parallel machines at station 2, i.e., $B_2 \geq S_2$. The processing times at each station are exponentially distributed with mean rates $\mu_i, i = 1, 2$. Buzacott and Shanthikumar (1993) (pages 205–206) and Perros (1994) (pages 64–65), among others, considered this problem. By forming the Markovian chain of this system, the random variable of interest is $N(t)$, the number of jobs which have been processed by the first station at time t and have not finished their processing at station 2 (at time t). $N(t), t \geq 0$ is a birth-death process with state space $s = \{0, 1, \dots, S_2, S_2 + 1, \dots, B_2, B_2 + 1, \dots, B_2 + S_1\}$. The birth rate is $\mu_1(\min S_1, B_2 + S_1 - v)$, whereas the death rate is $\mu_2(\min S_2, v)$, where, $v = 0, 1, \dots, S_2, S_2 + 1, \dots, B_2, B_2 + 1, \dots, B_2 + S_1$. Let $p(v)$ be the probability that there are v jobs in the second station including the jobs in the first station that are blocked. The steady-state (flow balance) equations associated with $p(v)$ are (see Perros, 1994, pp. 64–65):

$$\begin{aligned}
 S_1 \mu_1 p(0) &= \mu_2 p(1) \\
 (S_1 \mu_1 + v \mu_2) p(v) &= (v + 1) \mu_2 p(v + 1) + S_1 \mu_1 p(v - 1), \\
 &\quad \text{for } v = 1, 2, \dots, S_2 - 1, \\
 (S_1 \mu_1 + S_2 \mu_2) p(v) &= S_2 \mu_2 p(v + 1) + S_1 \mu_1 p(v - 1), \\
 &\quad \text{for } v = S_2, \dots, B_2, \\
 [(S_1 + B_2 - v)(\mu_1 + S_2 \mu_2)] p(v) &= S_2 \mu_2 p(v + 1) \\
 &\quad + [S_1 + B_2 - (v - 1)] \mu_1 p(v - 1), \\
 &\quad \text{for } v = B_2 + 1, \dots, B_2 + S_1 - 1, \\
 S_2 \mu_2 p(B_2 + S_1) &= \mu_1 p(B_2 + S_1 - 1).
 \end{aligned} \tag{2.91}$$

In steady state, the throughput of this system may be shown to be:

$$X = \sum_{v=0}^{S_2-1} v \mu_2 p(v) + S_2 \mu_2 \sum_{v=S_2}^{B_2+S_1} p(v) \quad (2.92)$$

$$X = \sum_{v=0}^{B_2} S_1 \mu_1 p(v) + \sum_{v=B_2+1}^{B_2+S_1} (B_2 + S_1 - v) \mu_1 p(v) \quad (2.93)$$

where $p(v)$, $v = 0, 1, \dots, S_2, S_2 + 1, \dots, B_2, B_2 + 1, \dots, B_2 + S_1$ are obtained from equations (2.91), above, by iteration:

$$p(v) = \begin{cases} \frac{S_1^v}{v!} \left(\frac{\mu_1}{\mu_2} \right)^v p(0), & v = 0, \dots, S_2 \\ \frac{S_1^v}{S_2! S_2^{v-S_2}} \left(\frac{\mu_1}{\mu_2} \right)^v p(0), & v = S_2 + 1, \dots, B_2 \\ \frac{S_1^{B_2} S_1!}{S_2! S_2^{v-S_2} (B_2 + S_1 - v)!} \left(\frac{\mu_1}{\mu_2} \right)^v p(0), & v = B_2 + 1, \dots, B_2 + S_1 \end{cases} \quad (2.94)$$

where probability $p(0)$ is obtained from the normalizing condition that the sum of all the steady-state probabilities is equal to 1.

As part of the development of a decomposition method (sub-section 2.5.3), Diamantidis, Papadopoulos and Heavey (2006) also solved the above problem exactly and the algorithm formulated by them is given below in sub-section 2.5.2.

This algorithm is available at the website associated with this book as special case of the 1184 2 (with the abbreviated name DECO-2) for $K = 2$ and in this case it gives the exact solution.

2.5.2 Alternative exact Markovian analysis of a two-station line with parallel machines at each station

The motivation for the development of this solution to the two-station multiple server line was to have available a building block for use in a decomposition approach to the solution of larger lines.

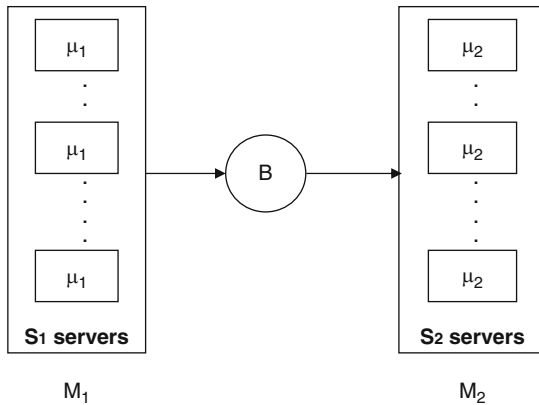


Fig. 2.18. A two-station, one-buffer production line with parallel machines at each station

Consider a system with two work-stations, which, for simplicity, are denoted by M_1 and M_2 (instead of WS_1 and WS_2 , respectively) as shown in Figure 2.18 consisting of S_1 and S_2 parallel machines, respectively. This system is used as the decomposition block in the decomposition approach given in sub-section 2.5.2. It is assumed that an inexhaustive supply of workpieces is available upstream of work-station 1, and an unlimited storage area is present downstream of work-station 2, viz., work-station 1 is never starved and work-station 2 is never blocked. Work-station i , $i = 1, 2$ consists of S_i reliable and identical machines, arranged in parallel and S_1 need not equal S_2 . Each parallel server has an exponentially distributed service time with mean $1/\mu_i$. The size or capacity of the intermediate buffer is denoted by B . The total storage capacity of the system is the physical storage of buffer B as well as the service positions at both work-stations 1 and 2. Therefore, the total storage capacity of the system, C , is $C = S_1 + S_2 + B$. Thus work-station 1 can be either *partially* or *fully blocked*. More specifically, if the current inventory of parts of the system (including those on the machines) equals $S_2 + B + 1$, then only one machine at work-station 1 is blocked and the remaining $S_1 - 1$ machines are not blocked. In this case, work-station 1 is partially blocked. If the storage of the system equals $S_1 + S_2 + B$, then all S_1 machines of the first work-station are blocked and, therefore, this work-station is fully blocked.

Because of the exponentially distributed service times, during the time interval $[t, t + dt]$ it is assumed only one event can occur at each work-station. Thus during the time interval $[t, t + dt]$ only one machine among the S_1 machines of work-station 1 can produce a part, or only one machine among the S_2 machines of work-station 2 can remove a part from buffer B . The total number of units in the system varies from 0 to $S_1 + S_2 + B$. It is straightforward that the total number of states is $S_1 + S_2 + B + 1$. Let $y = (c)$ denote the state of the system, where $c = 0, \dots, C$.

To solve this two-station system using exact Markovian analysis, the transition matrix must be derived. The following sub-section gives the transition equations. Then in sub-section 2.5.2 an algorithm for generating the transition matrix for any value C is presented.

Transition equations

The system states can be divided with respect to the storage level, c , into three sets: (i) lower boundary states; (ii) internal states; (iii) upper boundary states. It is further assumed that $S_1 \geq 1$, $S_2 \geq 1$ and $B \geq 0$.

Lower boundary state equation

The transition equation for state y with $c = 0$ (referred to as lower boundary state) has the following structure:

$$p_0 = (1 - S_1\mu_1)p_0 + \mu_2p_1. \quad (2.95)$$

Internal state equations

The transition equations for states $y = (c)$ with $0 < c < C$ can be sub-classified as follows:

Case 1: If $c > 0$ and $c < S_2$, then:

$$p_c = S_1\mu_1 p_{c-1} + (1 - S_1\mu_1 - c\mu_2)p_c + (c+1)\mu_2 p_{c+1}. \quad (2.96)$$

Case 2: If $c > S_2 - 1$ and $c < S_2 + B + 1$, then:

$$p_c = S_1\mu_1 p_{c-1} + (1 - S_1\mu_1 - S_2\mu_2)p_c + S_2\mu_2 p_{c+1}. \quad (2.97)$$

Case 3: If $c > S_2 + B$ and $c < C$, then:

$$p_c = S_2\mu_2 p_{c+1} + (C+1-c)\mu_1 p_{c-1} \\ + (1 - (C-c)\mu_1 - S_2\mu_2)p_c. \quad (2.98)$$

Upper boundary state equation

The state with storage level $c = C$ is called an upper boundary state. It holds that:

$$p_C = \mu_1 p_{C-1} + (1 - S_2\mu_2)p_C. \quad (2.99)$$

The algorithm for generating the transition matrix

Based on the above classification of the steady-state equations, an algorithm has been developed to generate the transition matrix of the two-station system (called decomposition block in the context of the decomposition approach, given below, in sub-section 2.5.2). Let P_{ij} , $i, j = 0, \dots, C$ be the element that is located in the i^{th} row and j^{th} column of the transition matrix P . The algorithm generates the transition probabilities in three stages: (i) transition probabilities of the lower boundary states (see Figure 2.19); (ii) transition probabilities of the internal states (Figure 2.20); (iii) transition probability of the upper boundary state (see Figure 2.19).

The transition matrix for the decomposition block can be generated using the algorithms presented in Figures 2.19 and 2.20. The Gaussian elimination method implemented in C++ is used to solve for the steady-state probabilities. The mean production rate (throughput) (X) of the decomposition block is calculated by using either of the following two formulas:

$$X = S_2\mu_2 \sum_{c=0}^{c=C} p_c - \mu_2 \sum_{c=0}^{c=S_2-1} (S_2 - c)p_c \quad (2.100)$$

or

$$X = S_1\mu_1 \sum_{c=0}^{c=C} p_c - \mu_1 \sum_{c=S_2+B+1}^{c=C} (c - S_2 - B)p_c. \quad (2.101)$$

{Lower boundary states}

```

 $P_{0,0} = 1 - S_1 \mu_1$ 
 $P_{0,1} = S_1 \mu_1$ 
for  $c = 2$  to  $C$  do
   $P_{0,c} = 0.0$ 
end for

```

{Upper boundary states}

```

 $P_{C,C-1} = S_2 \mu_2$ 
 $P_{C,C} = 1 - S_2 \mu_2$ 
for  $c = 0$  to  $C - 2$  do
   $P_{C,c} = 0.0$ 
end for

```

Fig. 2.19. Algorithm for generation of lower and upper boundary state transition probabilities

The expected in-process inventory (average storage level), \overline{WIP} , of the system can be calculated as follows (the reader is referred to Gershwin, 1994 and Helber, 1999)

$$\overline{WIP} = \sum_{c=0}^{c=C} c p_c. \quad (2.102)$$

The method for solving the decomposition block was validated using simulation. Sample results are given Table 2.11. For comparison purposes, a simulation model was developed in Arena V3.0 and the simulation results were found to be close enough to those obtained from the analytical model. Ninety-five percent confidence intervals were computed for any value B . The length of the simulation time is identical for all cases and equals 1100 time units.

For the experiments presented in Table 2.11, the processing times at both work-stations are assumed to be exponentially distributed with mean service rates, $\mu_1 = \mu_2 = 1$. In Table 2.11, the first column gives the number of parallel machines at the first work-station (S_1), the number of parallel machines at work-station 2 (S_2) and the buffer size, B . All these three values are represented by a vector (S_1, S_2, B) . The second column gives the throughput obtained by the numerical solution of the exact analytical algorithm proposed by Diamantidis, Papadopoulos and Heavey (2006), described above, while $X_{\text{algorithm}}$ and the third column gives the estimated 95% confidence intervals for the simulated mean production rates.

2.5.3 Approximate methods for large lines

Using the solution of the two-station line as a building block, the decomposition approach was applied by Diamantidis, Papadopoulos and Heavey (2006) to solve large-scale production lines consisting of K parallel-machine work-stations as those shown in Figure 2.21. Each work-station i , denoted for simplicity by M_i in the rest of

```

for  $i = 1$  to  $C - 1$  do
  for  $j = 0$  to  $j = C$  do
    if  $i > j$  and  $i - j = 1$  and  $i < S_2$  then
       $P_{i,j} = i\mu_2$ 
    end if
    if  $i > j$  and  $i - j = 1$  and  $i \geq S_2$  then
       $P_{i,j} = S_2\mu_2$ 
    end if
    if  $i = j$  and  $j < S_2$  and  $i < S_2 + B + 1$  then
       $P_{i,j} = 1 - S_1\mu_1 - j\mu_2$ 
    end if
    if  $i = j$  and  $j \geq S_2$  and  $i < S_2 + B + 1$  then
       $P_{i,j} = 1 - S_1\mu_1 - S_2\mu_2$ 
    end if
    if  $i = j$  and  $j \geq S_2$  and  $i \geq S_2 + B + 1$  then
       $K = C - i$ 
       $P_{i,j} = 1 - K\mu_1 - S_2\mu_2$ 
    end if
    if  $j > i$  and  $j - i = 1$  and  $i < S_2 + B + 1$  then
       $P_{i,j} = S_1\mu_1$ 
    end if
    if  $j > i$  and  $j - i = 1$  and  $i \geq S_2 + B + 1$  then
       $m = C - i$ 
       $P_{i,j} = m\mu_1$ 
    end if
    if  $i > j$  and  $i - j > 1$  then
       $P_{i,j} = 0.0$ 
    end if
    if  $j > i$  and  $j - i > 1$  then
       $P_{i,j} = 0.0$ 
    end if
  end for
end for

```

Fig. 2.20. Algorithm for generation of internal state transition probabilities

Table 2.11. Throughput of a two-work-station system with parallel machines

(S_1, S_2, B)	$X_{\text{algorithm}}$	95% CI for Simulated Throughput
(4,4,3)	3.52593	(3.48, 3.57)
(5,5,5)	4.54631	(4.47, 4.60)
(10,15,7)	9.99439	(9.80, 10.15)
(15,20,15)	14.99740	(14.75, 15.14)
(10,10,10)	9.45416	(9.25, 9.56)

this sub-section, consists of multiple identical reliable parallel machines with service rates $\mu_i, i = 1, \dots, K$ and intermediate buffers $B_i, i = 2, \dots, K$. The number of parallel machines at station i is $S_i, i = 1, \dots, K$, with each S_i an integer. Service times are exponentially distributed with mean $1/\mu_i$. It is also assumed that when any one of

the S_i parallel machines at work-station M_i completes a part, that part is placed in the buffer B_{i+1} downstream of the work-station immediately, provided the buffer is not full.

Markovian analysis of flow lines with moderate to large sized K is computationally expensive or impossible due to the enormous resulting state space (see Vidalis and Papadopoulos, 2001). Approximate methods are required to solve large systems. The work reported here uses the decomposition algorithm developed by Diamantidis, Papadopoulos and Heavey (2006). This decomposition method actually extends the work by Gershwin (1987) to solve flow lines with parallel servers at each work-station.

The solution approach for solving large lines with parallel machines at each work-station is as follows:

Following the derivation of the transition equations of the two-station system using exact Markovian analysis, an algorithm for generating the transition matrix for any two-station parallel system is developed. Thereafter, decomposition equations are derived using the well-known two-step methodology of obtaining the conservation flow equations and the flow rate idle time equations. Finally, a decomposition algorithm as outlined in Figure 2.22 was developed.

In the sequence, first, the decomposition equations are derived and then the decomposition algorithm is presented.

2.5.4 Derivation of the decomposition equations

In general, the decomposition method makes use of the four sets of equations (see Gershwin, 1994 where the decomposition method is described in great detail): (i) the conservation of flow equations; (ii) the flow rate idle equations; (iii) the resumption of flow equations; (iv) the interruption of flow equations. As the system analyzed here is reliable, only the first two sets of equations are used.

Conservation of flow equations

Let X_i^L be the mean production rate of the two-work-station, one-buffer sub-line L_i and X_i^u (X_i^d) be the mean production rate of the virtual upstream (downstream) pseudo work-station M_i^u (M_i^d), $i = 1, \dots, K - 1$. The mean production rate of each work-station M_i in the original line is denoted by X_i . The conservation of flow equations states that the production rates of all the two-work-station, one-buffer sub-systems L_i are the same.

Because the flow is conserved, it holds:

$$X_i^L = X_i^u = X_i^d = X_i, \quad i = 1, \dots, K - 1. \quad (2.103)$$

The flow rate idle time equations

Each virtual upstream pseudo work-station M_i^u of sub-line L_i , $i = 1, \dots, K - 1$, consists of S_i parallel machines, while each virtual downstream pseudo work-station M_i^d of line L_i consists of S_{i+1} parallel machines. The service times of the S_i parallel

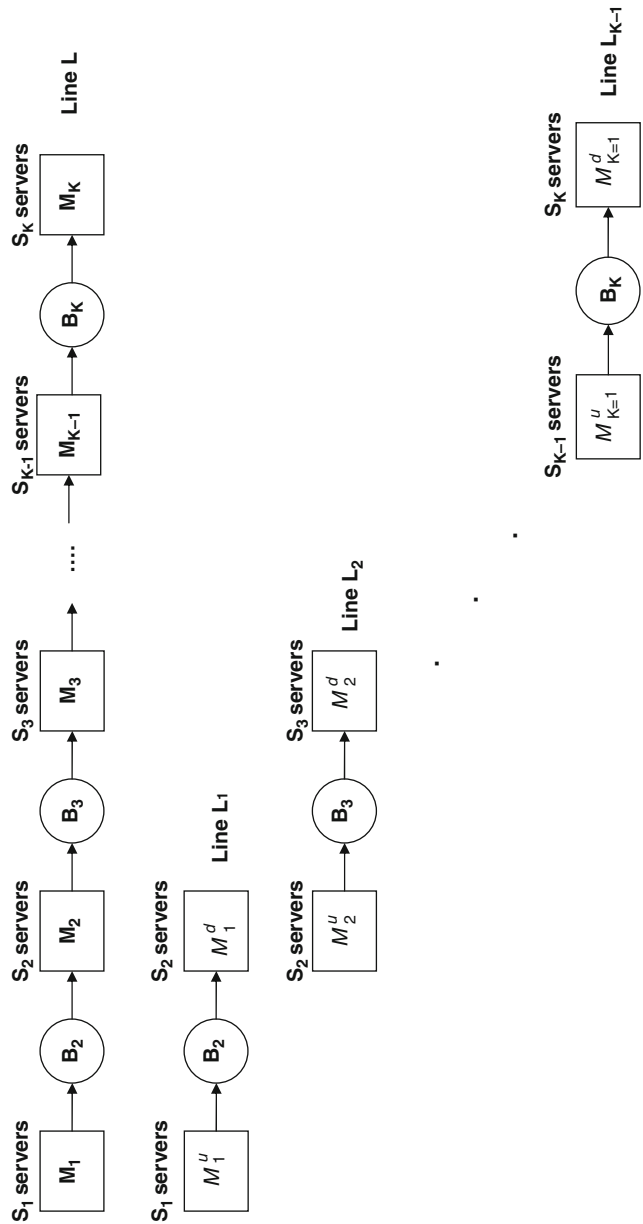


Fig. 2.21. Flow line with K parallel-machine work-stations, $K - 1$ intermediate buffers (Line L) and decomposition scheme (Lines L_1, \dots, L_{K-1})

{Step 1: Initialization}

for $i = 1$ to $K - 1$ **do**

$$\mu_i^u = \mu_i$$

$$\mu_i^d = \mu_{i+1}$$

ε = small positive number for terminating condition

end for

{Step 2: Calculate μ_i^u and μ_j^d }

for $i = 2$ to $K - 1$ **do**

Calculate μ_i^u using equation (2.116)

Evaluate the two-work-station, one buffer sub-line L_{i-1} , using the most recent values of μ_{i-1}^u and μ_{i-1}^d in the algorithm presented in sub-section 2.5.1.

end for

for $i = 2$ to $K - 1$ **do**

$$j = K - i$$

Calculate μ_j^d using equation (2.117)

Evaluate the two-work-station, one buffer sub-line L_{i+1} , using the most recent values of μ_{i+1}^u and μ_{i+1}^d in the algorithm presented in sub-section 2.5.1.

end for

{Step 3: Terminating Conditions}

if $|X_i^L - X_1^L| < \varepsilon, i = 2, \dots, K - 1$ **then**

GOTO Step 4

else

GOTO Step 2

end if

{Step 4: Output Results}

$$X = X_i^L, i = 1, \dots, K - 1$$

Fig. 2.22. Decomposition algorithm

machines of pseudo work-station M_i^u are exponentially distributed with mean $\frac{1}{\mu_i^u}$, while the service times of the S_{i+1} parallel machines of pseudo work-station M_i^d are also exponentially distributed with mean $\frac{1}{\mu_i^d}$, $i = 1, \dots, K - 1$. Also, define p_c^i to be the steady-state probability of state $y = (c)$ for sub-line L_i , where $c = 0, \dots, C_i$. Defining $C_i = S_i + S_{i+1} + B_i$ and taking into account equations (2.100) and (2.101), the mean production rate of work-stations M_{i-1}^d and M_i^u is given by the following formulae:

$$X_{i-1}^d = S_i \mu_{i-1}^d \sum_{c=0}^{c=C_{i-1}} p_c^{i-1} - \mu_{i-1}^d \sum_{c=0}^{c=S_i-1} (S_i - c) p_c^{i-1}, \quad i = 2, \dots, K \quad (2.104)$$

$$X_i^u = S_i \mu_i^u \sum_{c=0}^{c=C_i} p_c^i - \mu_i^u \sum_{c=S_{i+1}+B_i+1}^{c=C_i} (c - S_{i+1} - B_i) p_c^i, \quad i = 1, \dots, K - 1. \quad (2.105)$$

Rewriting equations (2.104) and (2.105):

$$X_{i-1}^d = \mu_{i-1}^d \left(S_i \sum_{c=0}^{c=C_{i-1}} p_c^{i-1} - \sum_{c=0}^{c=S_{i-1}-1} (S_i - c) p_c^{i-1} \right) \quad (2.106)$$

and

$$X_i^u = \mu_i^u \left(S_i \sum_{c=0}^{c=C_i} p_c^i - \sum_{c=S_{i+1}+B_i+1}^{c=C_i} (c - S_{i+1} - B_i) p_c^i \right). \quad (2.107)$$

It is straightforward to show that:

$$\sum_{c=0}^{c=C_i} p_c^i = \sum_{c=0}^{c=C_{i-1}} p_c^{i-1} = 1. \quad (2.108)$$

The blocking probability p_i^{bl} of each virtual two-work-station, one-buffer sub-line L_i is given by (derivation is omitted and the reader is addressed to Diamantidis, Papadopoulos and Heavey, 2006):

$$p_i^{bl} = \sum_{c=S_{i+1}+B_i+1}^{c=C_i} (c - S_{i+1} - B_i) p_c^i, \quad i = 1, \dots, K-1 \quad (2.109)$$

whereas the starvation probability p_{i-1}^{st} of each virtual two-work-station, one-buffer sub-line L_{i-1} is given by (derivation is omitted and the reader is addressed to Diamantidis, Papadopoulos and Heavey, 2006):

$$p_{i-1}^{st} = \sum_{c=0}^{c=S_{i-1}-1} (S_i - c) p_c^{i-1}, \quad i = 2, \dots, K. \quad (2.110)$$

Substituting equations (2.108) and (2.109) into equation (2.107), the mean production rate of the upstream work-station M_i^u is:

$$X_i^u = \mu_i^u (S_i - p_i^{bl}). \quad (2.111)$$

Similarly, substituting equations (2.108) and (2.110) into equation (2.106), the mean production rate of the downstream work-station M_{i-1}^d is:

$$X_{i-1}^d = \mu_{i-1}^d (S_i - p_{i-1}^{st}). \quad (2.112)$$

The mean production rate of work-station i , X_i , of the original production line L is given by:

$$X_i = \mu_i (S_i - p_{i-1}^{st} - p_i^{bl}). \quad (2.113)$$

Calculating probabilities p_i^{bl} and p_{i-1}^{st} from equations (2.111) and (2.112), respectively, one obtains:

$$p_i^{bl} = S_i - \frac{X_i^u}{\mu_i^u} \quad (2.114)$$

and

$$p_{i-1}^{st} = S_i - \frac{X_{i-1}^d}{\mu_{i-1}^d}. \quad (2.115)$$

Substituting equations (2.114) and (2.115) into equation (2.113) and taking into account conservation of flow in equation (2.103), the following two equations for calculating μ_i^u and μ_i^d can be derived:

$$\mu_i^u = \frac{1}{\frac{1}{\mu_i} + \frac{S_i}{X_{i-1}} - \frac{1}{\mu_{i-1}^d}}, \quad i = 2, \dots, K-1 \quad (2.116)$$

$$\mu_i^d = \frac{1}{\frac{1}{\mu_{i+1}} + \frac{S_{i+1}}{X_{i+1}} - \frac{1}{\mu_{i+1}^u}}, \quad i = K-2, \dots, 1. \quad (2.117)$$

Finally, because the virtual work-station M_1^u corresponds to the input work-station M_1 and the virtual work-station M_{K-1}^d corresponds to the output machine M_K of the original line L , the following boundary conditions are used:

$$\mu_1^u = \mu_1 \quad \text{and} \quad \mu_{K-1}^d = \mu_K. \quad (2.118)$$

2.5.5 The decomposition algorithm

Using the above derived equations, a decomposition algorithm shown in Figure 2.22 was developed. The ε value used in all the numerical examples given here was 0.00001.

2.5.6 Numerical results

In order to evaluate the performance and the accuracy of the proposed decomposition algorithm, several numerical experiments have been conducted by Diamantidis, Papadopoulos and Heavey (2006) for various configurations of production lines with parallel machines at each work-station. Here, a few representative sample numerical results are given. First, results for short lines of up to 7 stations are presented and compared to published results. Then, to illustrate the efficiency of the solution method, sample results for long production lines are presented.

Comparison with published exact results—Short lines

In Diamantidis, Papadopoulos and Heavey (2006), results for short lines with up to 7 work-stations were compared against those reported in Hillier and So (1989, 1995, 1996). Hillier and So applied exact Markovian analysis to calculate the throughput of small production lines with up to 7 work-stations in series. Here, in Table 2.12 and Table 2.13 sample results are given for lines with 5 stations, unbalanced lines (processing rates of machines at different stations are not the same but the processing rates of machines at any station are the same) and 3, 5 and 7 stations, balanced lines (all machines in all stations have the same processing rate) with different

Table 2.12. Comparison of results with Hillier and So (1996) – 5 work-stations

s	μ	X_{DECO}	X_{HS96}	% Error	Time
(1,1,1,1,4)	(2.0876, 2.7624, 3.0120, 3.4013, 0.2831)	1.0192	1.0240	0.469	0.01
(4,1,1,1,1)	(0.2831, 3.4013, 3.0120, 2.7624, 2.0876)	1.0192	1.0210	0.176	0.01
(1,1,1,4,1)	(2.0920, 2.8248, 3.2786, 0.2894, 2.4509)	0.9965	1.0120	1.532	0.01
(1,1,4,1,1)	(2.1739, 3.1347, 0.2905, 3.1347, 2.1739)	0.9913	1.0100	1.851	0.01
(2,1,1,1,3)	(0.6877, 2.9585, 2.9673, 3.1250, 0.3920)	0.9890	0.9790	1.021	0.01
(1,2,1,1,3)	(2.0533, 0.7710, 3.0581, 3.1250, 0.3892)	0.9745	0.9730	0.154	0.01
(1,1,2,3,1)	(1.9569, 2.6881, 0.7987, 3.2467, 0.3910)	0.9667	0.9690	0.237	0.01
(1,2,1,3,1)	(2.0408, 0.7686, 3.2154, 0.4103, 2.1691)	0.9514	0.9610	0.999	0.01
(1,1,1,1,6)	(3.0211, 4.0000, 4.4444, 5.1020, 0.2501)	1.4149	1.4130	0.134	0.01
(6,1,1,1,1)	(0.2501, 5.1020, 4.4444, 4.0000, 3.0211)	1.4149	1.4090	0.419	0.01

Table 2.13. Comparison of results with Hillier (1995) – 3, 5 and 7 work-stations

K	s	X_{DECO}	X_{HS95}	%Error	Time
3	(15,16,16)	13.5500	13.5400	0.074	0.05
3	(1,2,2)	0.8846	0.8873	0.304	0.01
3	(30,32,30)	27.7800	27.6800	0.361	0.27
5	(1,1,2,1,1)	0.5541	0.5638	1.720	0.02
5	(1,2,1,2,1)	0.6677	0.6637	0.603	0.02
5	(1,2,2,2,2)	0.8716	0.8752	0.411	0.02
7	(1,1,2,1,2,1,1)	0.5575	0.5613	0.677	0.05
7	(1,2,1,2,1,2,1)	0.6334	0.6320	0.222	0.05

servers allocation per station and zero buffer levels for all the intermediate buffers, respectively.

In all tables, columns labeled by vectors $s = (S_1, \dots, S_K)$, and $\mu = (\mu_1, \dots, \mu_K)$ denote, respectively, the server allocation and the mean service rate allocation at the respective work-stations of a production line with N work-stations. The column labeled X_{DECO} gives the estimated mean production rate using the proposed decomposition algorithm by Diamantidis, Papadopoulos and Heavey (2006), while the column labeled X_{HSXX} ($XX = 95, 96$) is the published results given in Hillier and So (1995, 1996). The percentage error between the results obtained from decomposition and those reported in Hillier (1995, 1996) is computed using the following formula:

$$\% \text{ Error} = \frac{|X_{\text{DECO}} - X_{\text{HSXX}}|}{X_{\text{HSXX}}} \times 100\%, \quad (2.119)$$

where XX denotes results reported in year 19XX.

Table 2.12 presents numerical results obtained for a production line consisting of 5 work-stations with all the buffer capacities equal to zero. Column 4 presents the mean production rate reported in Hillier (1996), (X_{HS96}). Column 5 (% Error) gives the percentage error between the results obtained from decomposition and those reported in Hillier (1996). Column 6 gives the time (in seconds) taken by

decomposition to estimate the throughput. The decomposition algorithm was run on a Pentium III at 450MHz with 256MB RAM.

Table 2.13 presents numerical results for production lines where the number of work-stations K are 3, 5 and 7. It is also assumed that the processing rates of all machines at each work-station are equal to 1 and that the buffer level for all the intermediate buffers is 0. The first column (K) gives the number of work-stations, whereas the fourth column gives the estimated throughput reported in Hillier (1995), (X_{HS95}). The fifth and sixth columns present the percentage error and the required time by decomposition to obtain the results, respectively.

Comparison with simulation results—Long lines

The decomposition algorithm proposed in Diamantidis, Papadopoulos and Heavey (2006) has also been used to estimate the throughput of large balanced and unbalanced production lines (up to 1000 and even more work-stations). This algorithm was developed and coded by Dr. Alexandros Diamantidis in C++. To our knowledge, there is no exact analytical method that can estimate the throughput of such large lines. Diamantidis, Papadopoulos and Heavey (2006) applied their algorithm and solved a large system with 1000 work-stations, each with 3 servers and 999 intermediate buffers each of buffer size equal to 10 buffer slots in approximately 50 minutes on a Pentium IV computer. For comparison purposes, Diamantidis, Papadopoulos and Heavey (2006) developed a simulation model in eM-Plant (http://www.ugs.com/products/tecnomatix/plant_design/em_plant.shtml) in order to compare the results obtained from the decomposition algorithm for large production lines. Recall that this decomposition algorithm for $K = 2$ stations specifies the two-station line, with parallel machines at each work-station and an intermediate buffer, which is solved exactly and the throughput of the system is calculated.

S_i , and μ_i , $i = 1, \dots, K$, denote the number of parallel machines at work-station i and the service rate of each one of the S_i parallel machines at work-station i , respectively. B_i , $i = 2, \dots, K$, denote the storage capacity of the buffer located in front of work-station i , $i = 2, \dots, K$.

Table 2.14 presents configurations for 12 sample production lines varying from 10 to 120 stations in steps of 10. The results given for each example in Table 2.15 are: (i) throughput of the system calculated using simulation (X_{SIM}); (ii) throughput of the system calculated using the decomposition algorithm (X_{DECO}); (iii) average inventory for the system calculated using simulation (\bar{c}_{SIM}); (iv) average inventory for the system calculated using the decomposition algorithm (\bar{c}_{DECO}). Table 2.15 also presents 95% confidence intervals calculated for X_{SIM} and \bar{c}_{SIM} . The % Error for X was calculated using equation (2.119) with the % Error for \bar{c} calculated similarly. The computing time, in seconds, to execute the decomposition algorithm is given in the two columns labeled “ X -Time” and “ \bar{c} -Time,” respectively. These columns give the time to calculate the throughput and the average inventory, respectively. The computing time, in seconds, to execute the simulation experiments is given in the last column. The simulation model was run for 20,000 units/customers before statistics

Table 2.14. Sample configurations for long lines

Production Line 1	Production Line 2	Production Line 3
$K = 10$ $S_i = 2, i = 1, \dots, 4$ $S_5 = S_6 = 3$ $S_i = 2, i = 7, \dots, 10$ $B_i = 3, i = 1, \dots, 9$ $\mu_i = 1, i = 1, \dots, 10$	$K = 20$ $S_i = 3, i = 1, \dots, 7$ $S_j = 4, j = 8, \dots, 15$ $S_k = 3, k = 16, \dots, 20$ $B_i = 4, i = 1, \dots, 19$ $\mu_i = 1, i = 1, \dots, 20$	$K = 30$, $S_i = 3, i = 1, \dots, 10$ $S_j = 4, j = 11, \dots, 20$ $S_k = 3, k = 21, \dots, 30$ $B_i = 4, i = 1, \dots, 29$ $\mu_i = 1, i = 1, \dots, 30$
Production Line 4	Production Line 5	Production Line 6
$K = 40$ $S_i = 2, i = 1, \dots, 15$ $S_j = 3, j = 16, \dots, 25$ $S_k = 2, k = 26, \dots, 40$ $B_i = 3, i = 1, \dots, 49$ $\mu_i = 1, i = 1, \dots, 50$	$K = 50$ $S_i = 2, i = 1, \dots, 10$ $S_j = 1, j = 11, \dots, 40$ $S_k = 2, k = 41, \dots, 50$ $B_i = 2, i = 1, \dots, 49$ $\mu_i = 1, i = 1, \dots, 50$	$K = 60$ $S_i = 4, i = 1, \dots, 20$ $S_j = 5, j = 21, \dots, 40$ $S_k = 4, k = 41, \dots, 60$ $B_i = 4, i = 1, \dots, 59$ $\mu_i = 1, i = 1, \dots, 60$
Production Line 7	Production Line 8	Production Line 9
$K = 70$ $S_i = 3, i = 1, \dots, 25$ $S_j = 2, j = 26, \dots, 45$ $S_k = 3, k = 46, \dots, 70$ $B_i = 2, i = 1, \dots, 69$ $\mu_i = 1, i = 1, \dots, 70$	$K = 80$ $S_i = 2, i = 1, \dots, 30$ $S_j = 3, j = 31, \dots, 50$ $S_k = 2, k = 51, \dots, 80$ $B_i = 2, i = 1, \dots, 79$ $\mu_i = 1, i = 1, \dots, 80$	$K = 90$ $S_i = 2, i = 1, \dots, 30$ $S_j = 3, j = 31, \dots, 60$ $S_k = 2, k = 61, \dots, 90$ $B_i = 2, i = 1, \dots, 89$ $\mu_i = 1, i = 1, \dots, 90$
Production Line 10	Production Line 11	Production Line 12
$K = 100$ $S_i = 2, i = 1, \dots, 40$ $S_k = 3, k = 41, \dots, 60$ $S_m = 2, m = 61, \dots, 100$ $B_i = 2, i = 1, \dots, 99$ $\mu_i = 1, i = 1, \dots, 100$	$K = 110$ $S_i = 5, i = 1, \dots, 40$ $S_j = 6, j = 41, \dots, 70$ $S_k = 5, i = 71, \dots, 110$ $B_i = 4, i = 1, \dots, 110$ $\mu_i = 1, i = 1, \dots, 110$	$K = 120$ $S_i = 3, i = 1, \dots, 50$ $S_j = 4, i = 51, \dots, 70$ $S_k = 3, i = 71, \dots, 120$ $B_i = 3, i = 1, \dots, 120$ $\mu_i = 1, i = 1, \dots, 120$

were collected. The batch means method was used to collect 30 independent samples within a single run. A batch size of 5000 units/customers was used.

From examination of Table 2.15 it can be observed that the maximum error for the throughput is 1.72%. The accuracy of the decomposition algorithm for average inventory is not as good with a maximum error of 16.27% observable in Table 2.15. The convergence of the algorithm was found, for the majority of cases, to be very fast. However, the convergence speed can vary considerably and is system dependent, as can be observed in Table 2.15 where Line # 9 and 10 took approximately 50% of the time it took to obtain results using simulation. For all the results of the decomposition algorithm, a Pentium III at 450MHz with 256MB RAM was used. The simulation experiments were carried out on a Pentium IV at 2992MHz with 1000MB of RAM.

Table 2.15. Sample numerical results for long lines

Line #	Throughput			Average Inventory			Decomposition Time		Simulation Time		
	\bar{X}_{SIM}	\bar{X}_{DECO}	% Error	95% CI	\bar{c}_{SIM}	\bar{c}_{DECO}	% Error	95% CI		X-Time	\bar{c} -Time
1	1.5242	1.5423	1.1739	1.5198–1.5285	32.95679	33.4985	1.6171	30.7799–35.1336	0.09	0.39	87.46
2	2.3884	2.3808	0.3183	2.3805–2.3961	87.02922	77.5526	12.2196	83.4917–90.5666	0.61	0.66	136.83
3	2.3315	2.3533	0.9269	2.3402–2.3506	143.4842	139.6802	2.7234	138.9421–148.0263	0.22	0.22	183.85
4	1.4387	1.4447	0.4184	1.4353–1.4419	135.6619	130.001	4.3545	131.2453–140.0785	54.65	56.35	252.26
5	0.6092	0.6036	0.9335	0.6081–0.6102	101.4165	116.9982	13.3179	97.5978–105.2352	0.98	0.99	303.57
6	3.1274	3.1662	1.2250	3.1204–3.1343	347.8995	321.6019	8.1771	340.8268–354.9722	1.43	1.26	366.95
7	1.3440	1.3412	0.2071	1.3407–1.3472	214.3427	255.9992	16.2721	208.7912–219.8942	13.46	1.27	437.96
8	1.3311	1.3389	0.5814	1.3283–1.3339	239.463	215.59	11.0733	233.5951–245.3308	42.24	62.47	477.1
9	1.3319	1.339	0.5311	1.329–1.3347	262.4053	230.581	13.8018	256.2628–268.5477	244.69	249.64	562.71
10	1.3241	1.3366	0.9315	1.3212–1.327	279.8781	274.1059	2.1058	273.5344–286.2217	260.68	266.6	616.71
11	3.9448	4.0066	1.5418	3.9357–3.9538	768.0851	701.1589	9.5451	757.5761–778.5941	2.64	1.27	704.81
12	2.1926	2.2309	1.7175	2.1883–2.1968	529.9708	503.4035	5.2775	521.2414–538.7001	2.97	3.02	752.33

Table 2.16. Configurations for longer lines

Production Line 13	Production Line 14	Production Line 15
$K = 200$	$K = 300$	$K = 400$
$S_i = 3, i = 1, \dots, 50$	$S_i = 3, i = 1, \dots, 50$	$S_i = 2, i = 1, \dots, 100$
$S_i = 2, i = 51, \dots, 100$	$S_i = 1, i = 51, \dots, 100$	$S_i = 3, i = 101, \dots, 200$
$S_i = 4, i = 101, \dots, 150$	$S_i = 4, i = 101, \dots, 150$	$S_i = 4, i = 201, \dots, 300$
$S_i = 1, i = 151, \dots, 200$	$S_i = 2, i = 151, \dots, 200$	$S_i = 1, i = 301, \dots, 400$
$B_i = 2, i = 1, \dots, 199$	$S_i = 3, i = 201, \dots, 300$	$B_i = 2, i = 1, \dots, 399$
$\mu_i = 1, i = 1, \dots, 200$	$B_i = 3, i = 1, \dots, 299$	$\mu_i = 1, i = 1, \dots, 400$
	$\mu_i = 1, i = 1, \dots, 300$	
Production Line 16	Production Line 17	Production Line 18
$K = 500$	$K = 600$	$K = 700$
$S_i = 3, i = 1, \dots, 150$	$S_i = 2, i = 1, \dots, 200$	$S_i = 4, i = 1, \dots, 250$
$S_i = 4, i = 151, \dots, 300$	$S_i = 4, i = 201, \dots, 400$	$S_i = 2, i = 251, \dots, 450$
$S_i = 2, i = 301, \dots, 500$	$S_i = 3, i = 401, \dots, 600$	$S_i = 4, i = 451, \dots, 700$
$B_i = 3, i = 1, \dots, 499$	$B_i = 2, i = 1, \dots, 599$	$B_i = 3, i = 1, \dots, 699$
$\mu_i = 1, i = 1, \dots, 500$	$\mu_i = 1, i = 1, \dots, 600$	$\mu_i = 1, i = 1, \dots, 700$
Production Line 19	Production Line 20	Production Line 21
$K = 800$	$K = 900$	$K = 1000$
$S_i = 4, i = 1, \dots, 200$	$S_i = 4, i = 1, \dots, 400$	$S_i = 4, i = 1, \dots, 400$
$S_i = 3, i = 201, \dots, 400$	$S_i = 3, i = 401, \dots, 500$	$S_i = 3, i = 401, \dots, 600$
$S_i = 2, i = 401, \dots, 600$	$S_i = 4, i = 501, \dots, 900$	$S_i = 4, i = 601, \dots, 1000$
$S_i = 3, i = 601, \dots, 800$	$B_i = 3, i = 1, \dots, 899$	$B_i = 3, i = 1, \dots, 999$
$B_i = 3, i = 1, \dots, 799$	$\mu_i = 1, i = 1, \dots, 900$	$\mu_i = 1, i = 1, \dots, 1000$
$\mu_i = 1, i = 1, \dots, 800$		

In Table 2.17 throughput results are given for the configurations shown in Table 2.16 which include even longer production lines (with $K = 200(100)1000$ workstations). As it can be seen from Table 2.17, the maximum error was found to be 2.5%. The parameters s , number of parallel stations, n , the buffer sizes, and μ , the mean service times, vary arbitrarily, so as to illustrate the versatility of the algorithm. Run times, in seconds, for the decomposition algorithm and the simulation model are given in the last two columns. From Table 2.17 it can be noted that X_{DECO} lies outside the 95% CI for Line # 16–21. In general it was found that X_{DECO} was outside the 95% CI for configurations with % Error greater than 1.00.

The numerical results presented in Table 2.15 and in Table 2.17 indicate that the decomposition algorithm is very accurate. The average percentage error of the throughput obtained from the proposed decomposition algorithm and simulation for lines with up to 100 stations is less than 1%, whereas the results presented for lines with up to 1000 stations indicate that the percentage error is less than 2.5%. The convergence of the algorithm is very fast and reliable. Diamantidis, Papadopoulos and Heavey (2006) claim that they have not found a case in which the algorithm does not converge.

Table 2.17. Numerical results for longer lines

Line #	X_{SIM}	X_{DECO}	% Error	95% CI	Decomposition Time	Simulation Time
13	0.6047	0.6014	0.5444	0.6036–0.6058	7.58	1340.72
14	0.6620	0.6678	0.8688	0.6610–0.6631	15.05	1478.35
15	0.6001	0.6004	0.0546	0.5994–0.6008	131.77	3104.08
16	1.3984	1.4287	2.1651	1.3963–1.4012	369.16	3673.55
17	1.3089	1.3335	1.8763	1.3073–1.3105	347.08	3188.14
18	1.3988	1.4287	2.1411	1.3956–1.4012	361.09	4618.73
19	1.3982	1.4287	2.1790	1.3960–1.4003	571.55	5626.19
20	2.1802	2.2292	2.2470	2.1766–2.1838	239.97	5887.03
21	2.1759	2.2287	2.4276	2.1722–2.1796	804.60	7162.45

The above algorithm is available at the website associated with this book, with abbreviated name DECO-2. As noted above if K , the number of stations is equal to 2, the exact numerical solution to the two-station production line with identical perfectly reliable parallel machines at each station may be obtained. In addition, if the number of machines at each station is set equal to 1, the authors have shown that the results obtained for large production lines with single machine stations, exponential service times, perfectly reliable machines and intermediate buffers of finite capacity replicate the decomposition equations originally given by Gershwin (1994).

2.6 Simulation Modeling

Simulation of production lines is a powerful tool in obtaining the performance measures where analytical methods are either difficult or impossible to use. In the past, simulation was often considered to be an expensive and time consuming approach to the solution of system type problems. However, with the increase of computer power and the availability of special-purpose simulation languages, such constraints are less severe. Usually, in simulation studies of production lines what is technically involved is Monte Carlo simulation because of the inherent stochastic variability of these systems. The combination of simulation studies with analytical studies is probably the way of the future in the design of production lines.

Ideally, the production line analyst and designer requires a discrete event simulation package with Monte Carlo simulation capabilities, graphical and other output reporting facilities together with a relatively easy method for the statistical assessment of results. In simulation modeling, the modeler must specify very carefully how the production line is meant to operate and the various disciplines and rules which are involved. The basis of discrete event simulation is that the system state at any time t is stored and that the state only changes when a particular event occurs. The specification of the state of the system depends on the detail required by the modeler in respect to the performance characteristics of the system. In all simulation studies there is a need to consider the time horizon of the process under investigation.

Short term system performance analysis requires that data be taken from the simulation during a short time horizon. On the other hand, steady-state simulation models are appropriate for the analysis of systems which in theory could run indefinitely. Usually, in production line modeling the modeler is interested in steady-state behavior of the system by which time the precise initial state of the system has little impact. It is normal in these cases to have a “warm-up” period before recording data for the calculation of the steady-state behavior. Graphical output of the performance parameters of the system can be extremely useful in determining when the warm-up period is ended. In some simulation packages it is possible to specify in advance the time to allow the system to settle down. The appropriateness of this time may in fact be checked from the results of the simulation. Finally, it is usual to place confidence limits on the values of the performance parameters of the system, based on certain assumptions, and such limits may usually be incorporated into modern simulation models.

As an illustration of the power of simulation, Arena, a simulation software package available to the authors, has been used to model a system of the type depicted in Figure 2.23, with the following characteristics:

- The line consists of $K = 4$ work-stations with identical parallel machines at each work-station. The number of machines at station $i, i = 1, \dots, 4$ are 3, 2, 2, 3, respectively. Service or processing times are exponentially distributed and the mean service rates of the identical machines at each work-station are $\mu_i = 1, i = 1, \dots, 4$. Thus, the probability that a service is completed on a machine at station i in a time interval Δt is $\mu_i \Delta t$.
- All machines are assumed to be perfectly reliable.
- The interstation buffers $B_i, i = 2, 3, 4$ have capacities of 4, 2, 4, respectively.
- All products produced conform to specifications.
- Transfer times between stations and buffers and between buffers and stations are considered negligible.
- Any particular machine may be blocked after service due to the finite capacities of the buffers excluding the last set of parallel machines.
- Arrangements are made to ensure that the first station is always busy, i.e., never starved or it is saturated; any machine at any other station may be starved.
- No machine of any of the group of machines at any particular station is given priority in relation to being unblocked when unblocking occurs. Likewise in relation to the resumption of production at a work-station following the removal of starvation.

The following performance measures of the line were determined:

- Throughput (jobs exiting from the production line per unit time).
- Utilization of each work-station (the limit of the time average of the number of busy machines over time divided by the total number of machines in the station).
- Average buffer level for each intermediate buffer.
- Average work-in-process, \overline{WIP} , excluding the buffer before the first station.
- Average job waiting time at each of the intermediate buffers.

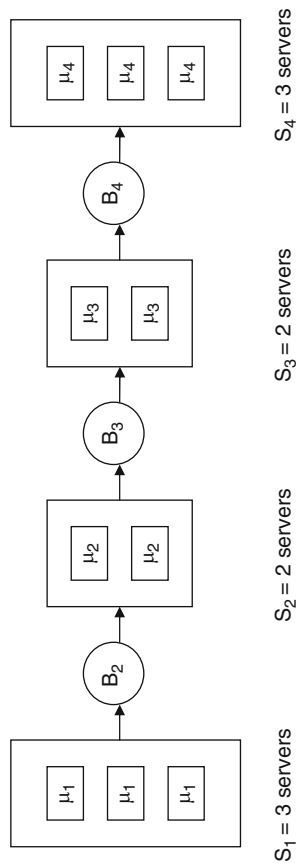


Fig. 2.23. A production line with four stations with parallel reliable machines at each station and three intermediate buffers

To ensure saturation of the first station of the line, the capacity of the queue in front of the first station was set at twenty (20) units and batches arrive at sizes of five (5), and the arrival rate into the system, λ , was taken to be greater than $3\mu_1$. Finally, a warm-up period of 100 minutes was specified to ensure that steady-state conditions were obtained. (More details of this Arena simulation model and numerical results are given in Appendix E.)

It may be of interest to discuss briefly possible extensions of the relatively simple simulation model developed above. Clearly, the number of stations and the number of parallel machines at each station could be modified. Buffer capacities could be changed, and the restriction of identical machines at each work-station could be removed which would impact on the scheduling rules. Unreliable machines could be incorporated. Processing times, repair times, times to failures and transport times from stations to buffers and from buffers to stations may be modeled using the Monte Carlo simulation. There is no need to confine the distributions to be exponential as Erlang, phase-type, normal, uniform or deterministic distributions may be modeled using Monte Carlo simulation. The production of defective items at any station could be incorporated into the model via a feedback mechanism for rework if necessary to earlier stations.

2.7 General Comment

The reader will note that the Markovian method gives an exact evaluation of the state probabilities from which an exact evaluation of the throughput may be obtained, whereas the other three methods (aggregation, decomposition and expansion) give approximate results for the throughput. Clearly, the limitation in using Markovian methods is the smaller number of states which may be handled effectively in contrast to the approximate methods at the expense of accuracy. The existence of parallel-machine stations introduced complications on two fronts, viz., significantly increasing the number of states and the difficulties of such elements being incorporated in the early classical decomposition methods. The expansion method gave early hope that some of the problems with the modeling of parallel machines would be reduced, but issues relating to accuracy of results still remain. It appears that the results reported in Section 2.5 overcome a number of the problems associated with earlier attempts to use decomposition methods in systems with parallel structures. In principle, it is possible to derive the states for Markovian type of solution for such systems, but unfortunately this would be computationally very inefficient for any realistic production lines with parallel-machine stations. All methods presented are, of course, in principle applicable to unreliable as well as perfectly reliable machines.

2.8 Related Bibliography

Researchers and practitioners alike should appreciate that there is a very rich literature applicable to the general area of the performance evaluation of production lines. Here an attempt is made to highlight the main strands of thought in this area. It should

be noted that work initiated for application areas quite diverse from manufacturing has been found to be fruitful when applied to the analysis of production lines. Cases in point are analyses originally oriented toward computer performance modeling and communication networks have subsequently given insights into problems germane to the analysis of production lines. Basically, the mathematical underlined theory of production line analysis is queueing theory, in particular, queueing networks with blocking. An exceptional reference in this area is the book by Perros (1994). However, care must be taken to ensure the validity of the model of the production system in that for example blocking in a communication system tends to occur before the service starts, whereas in a production line blocking occurs after the service has been completed. What is offered below is a classification by the authors of what they believe to be significant and somewhat distinct areas of research of value in the analysis of the performance of production lines. As far as Markovian analysis approaches are concerned, some five areas of work have been identified and are described below. It should be noted that there is nothing unique about this categorization and indeed some authorities might well question the relative influence accorded to the work of particular researchers.

The exact solution of small production lines was initiated by Hunt (1956) followed by Buzacott (1972), Gershwin and Berman (1981) and Gershwin and Schick (1983), among others. Solutions were obtained for two/three stations with limited inter-station buffers, and methods of solution used included matrix recursive and matrix geometric methods applied to the underlying Markov chains. Initially, exponentially distributed processing times were only considered, but the work of Buzacott and Kostelski (1987) extended the distribution of processing times to phase-type distributions.

Altiok (1997) in a seminal work summarized and developed the earlier work by Altiok and Ranjan (1989), Buzacott and Kostelski (1987) and Perros (1994), among others, and brought phase-type modeling to its present position. Exact analysis of small-scale production lines with any type of processing and repair time distributions may be undertaken. Arising out of Altiok's work it is possible to perform approximate analysis of larger systems with any general distributions of processing and repair times by the approximation of these distributions by phase-type distributions by matching their first two or three moments.

When faced with the analysis of relatively large production lines, there is a need for efficient computational procedures due essentially to the large number of associated states of the underlying Markov chain of such systems. Hillier and Boling (1967) developed a numerical approach for solving reliable exponential and Erlang production lines. Papadopoulos and O'Kelly (1989), Papadopoulos, Heavey and O'Kelly (1989, 1990) and Heavey, Papadopoulos and Browne (1993) further developed this work by producing efficient numerical algorithms for generating the transition matrices for reliable and unreliable production lines with exponential and Erlang processing and repair time distributions and efficient solution methods. Further extensions in this area are included in the book of Altiok (1997), as noted above, by using the mixed generalized exponential distributions (phase-type distributions). The algorithm included at the website associated with this book with abbreviated

name MARKOV for the generation of the transition matrix and the solution of the associated steady-state Markov equations is based on the work of Papadopoulos, O'Kelly and Heavey.

In contrast to continuous parameter discrete state Markov process analysis of production lines, Muth (1984) introduced the concept of the holding time model where the focus is on the three possible states of each station, viz., the station is idle, busy or blocked. Alkaff and Muth (1987) extended Muth's model to solve K -station production lines with an arbitrary number of stations. A major advantage of the holding time model is that the number of separate non-linear equations that have to be solved is significantly reduced in comparison to the Markovian situation. The price paid for this reduction is the need to solve non-linear equations that are of the form of a fixed point problem. Holding time models can accommodate Erlang and phase-type distributions more readily than can Markovian methods again because of the reduction in the number of states. However, the holding time model cannot accommodate intermediate buffers of non-zero capacity.

It is of great assistance to designers to have simple closed form formulae to determine the throughput of production lines. A number of such formulae have been developed based on insights from general queueing theory, considerations and sometimes curve-fitting. Hunt (1956) was an early developer of such a closed form expression. Makino (1964), Muth (1984) and Muth (1987) offered formulas for the exponential and two-phase Erlang and distribution-free cases with no intermediate buffers between successive stations. Freeman (1968) and Anderson and Moodie (1969) obtained empirical formulas for utilization of the production line, based on regression analysis of various sets of simulation data. Knott (1970) offered a formula based on theoretical and intuitive reasoning. Blumenfeld (1990) extended Muth's formula for throughput of a production line with variable processing times and buffers of finite capacities. Haydon (1973) dealt with approximations in his Ph.D. dissertation and he provided approximate throughput formulae that perform quite satisfactorily. Papadopoulos (1996) using Muth's holding time model developed an analytical formula for the throughput of a K -station production line with no intermediate buffers and exponential processing times which may be different at the various stations of the line. A particular simpler formula was developed for the balanced line.

The limitations of seeking exact solutions to production line problems are related to problems arising from the number of states of such systems and the difficulties associated with a numerical approach. Thus, there has been considerable interest in developing approximate methods of analysis. Most approximate methods are based on decomposition and an essential element of this approach is that the sub-lines used have exact solutions. Decomposition methods are approximations as the sub-lines used are simpler than the original line and the equations used to develop the parameters may also be approximated to facilitate the numerical analysis. Earlier work on decomposition methods include Zimmern (1956), Sevast'yanov (1962) and Hillier and Boling (1967). Queueing networks with blocking were decomposed by Caseau and Pujolle (1979), Takahashi et al. (1980), and Boxma and Konheim (1981). Altioik and Perros and their teams have made significant contributions by working on decomposition to solve large systems with exponential and phase-type distributed

processing times. This work is reported in papers including Altioik (1982), Perros and Altioik (1986), Jun and Perros (1987), Brandwajn and Jow (1988), Altioik (1989), Altioik and Ranjan (1989) and Gun and Makowski (1989). Excellent expositions of this work are given in the book of Altioik (1997). Gershwin (1987) in a well-known article offered an efficient decomposition method for the approximate evaluation of tandem queues with finite intermediate buffers and blocking. Dallery, David and Xie (1988) improved the convergence of Gershwin's algorithm. An excellent review of flow line models is given in Dallery and Gershwin (1992), and the decomposition approaches are treated in detail in the book of Gershwin (1994). Decomposition models of various types of manufacturing systems are also included in the seminal work of Buzacott and Shanthikumar (1993). Dallery and Frein (1993) classified the various decomposition methods for solving production lines into one of three classes according to the sets of decomposition equations used by the various authors.

Many papers concerned with the analysis of production lines have reported results on simulation studies. It is virtually impossible to give an adequate review of such papers from the perspective of the use of simulation method in the determination of production line performance. Nevertheless, there are a number of books and research papers which are certainly worth further detailed study and investigation by analysts specifically interested in simulation. These include the books by Altioik and Melamed (2001), Kouikoglou and Phillis (2001), Law and Kelton (2000), Guide to Arena Standard Edition by Systems Modeling Corporation (1999), Banks et al. (1999), Kelton et al. (1998), Benson (1996), Khoshnevis (1994), Papadopoulos et al. (1993), Brateley et al. (1987), Pritsker (1986) and Fishman (1973), among others.

Decomposition techniques have also been applied not only to manufacturing systems but also to computer systems (see Perros, 1994 and many references therein), among others, and to more general manufacturing systems. For example, Tempelmeier and Burger (2001a) examined non-homogeneous asynchronous flow production systems and presented an analytical approximation for the performance of such systems. They assumed generally distributed stochastic processing times as well as breakdowns and imperfect production. The proposed approximation was based on the decomposition of an K -station-line into $(K - 1)$ two-station-lines that were analyzed using a $GI/G/1/N_{\max}$ queueing model. They also presented numerical comparisons with exact and simulation results which indicated that the procedure provides accurate results. In Kuhn (2003) an analytical approach was given for performance evaluation of an automated flow line system which considers the dependency between the production and the repair system. The proposed model and solution approach may be used in the initial design phase as well as during a redesign process in order to evaluate various configurations of the production and repair systems.

Tolio and Matta (1998) presented an elegant decomposition approach for the performance evaluation of automated flow lines with multiple failure modes. The decomposition block that was used in their analysis was solved exactly by a method that is independent of the buffer size. An extension of the decomposition approach for the performance evaluation of a flow line with linear flow of material and two part types was presented by Nemec (1999). A different efficient decomposition analysis

for serial flow lines with two part types, deterministic identical processing times and multiple failure modes was proposed by Colledani, Matta and Tolio (2003). Flow lines with single machine work-stations and non-linear flow of material are examined in Helber (1999), where a detailed analysis of flow lines with split and merge operations is presented, Gershwin et al. (2001), Helber and Mehrrens (2003), Tan (2001) and Helber and Jusic (2004).

Tolio, Matta and Gershwin (2002) presented an analytical method for the performance evaluation of production lines with two unreliable machines and one intermediate buffer of finite capacity. Each machine can fail in more than one way.

Levantesi, Matta and Tolio (1999a,b) developed an efficient decomposition method for the performance evaluation of production lines with exponential processing times, multiple failure modes and finite buffer capacities. The different types of failures are distributed according to different exponential distributions as are the times to repair.

Levantesi, Matta and Tolio (2003) provided an approximate analytical method for the performance evaluation of asynchronous production lines with deterministic processing times, multiple failure modes and finite buffer capacity. In their analysis, the authors approximated the discrete flow of parts by a continuous flow of material.

Literature is relatively scarce on the analysis of flow lines with multiple identical parallel-machine work-stations. Friedman (1965) presented a reduction method that reduces a queueing system with parallel-machine work-stations to corresponding problems for a system of fewer stages. It was also assumed that for any sequence of customer arrival times, the time spent in the system was independent of the order of stages. Forestier (1980) examined automated flow lines where each station consists of two parallel machines. Dubois and Forestier (1982) considered similar systems using Markovian analysis. Iyama and Ito (1987) considered a flow line where some work-stations have different numbers of parallel machines and unequal service rates. They presented the effects of server allocation on the maximum average production rate by using a Markovian model.

The exact solution of the two-station production line with the first station saturated is based on queueing theory and a good exposition of this analysis may be found in the book by Perros (1994), in the book by Buzacott and Shanthikumar (1993) and in the book by Neuts (1981), among others. Details of the generation of the associated conservative matrix, A , and a method for the calculation of the throughput of such systems based on the elements of A are given in the paper by Vidalis and Papadopoulos (2001). In addition, the recursive relationship for the number of states of a general production line with $K \geq 2$ parallel stations is derived in this paper. With respect to the approximate solutions of larger systems there are a few research studies of interest. These include the book by Buzacott and Shanthikumar (1993), where an iterative procedure is applied to calculate the throughput of the long line using the solution to the two-station line described above. In the paper by Jain and MacGregor Smith (1994), the expansion method was used to approximate the performance measures of each parallel station of the production line. In this paper, apart from the series system, merge and splitting topologies were also analyzed.

Another paper of major interest is that by Patchong and Willaeyts (2001), where each set of parallel machines is replaced by an equivalent single machine at each station of the production line. Then, existing methods may be used to derive the performance measures of the original system. A similar approximation method was applied by Jeong and Kim (1999) for performance analysis of assembly/disassembly systems with parallel machine stations. Earlier, Caseau and Pujolle (1979) derived the throughput of some specialized telecommunication models using repeated trials methods.

In van Dijk and van der Wal (1989) computationally attractive lower and upper bounds for finite multi-server exponential tandem queues were presented. A proof of the bounds and related monotonicity results were also presented, which were based on Markov reward theory. Gosavi and MacGregor Smith (1995) developed computationally efficient bounds and approximations for the performance measures of series parallel queueing networks. They approximated analytically the throughput of a system with two tandem exponential queues and extended their analysis to elementary merge and split queueing networks.

Ancelin and Semery (1987) described a method that replaces each parallel-machine work-station by an equivalent single machine work-station. The processing rate of the equivalent work-station equals the sum of the processing rates of all parallel machines in the work-station. The failure rate and repair rate of the equivalent work-station are given by a formula which incorporates the failure and repair parameters of the parallel machines in the work-station.

Burman (1995) applied a similar method that replaces each parallel server work-station by a single equivalent work-station for the case of continuous flow of material. The author assumed that the equivalent work-station has a maximum processing rate which equals the sum of the processing rates of the parallel machines. The failure and repair parameters of the equivalent work-station are calculated by using the assumption that all parallel machines at a specific work-station operate independently.

Cheah and MacGregor Smith (1994) showed how a $M/G/C/C$ state dependent queueing model is embodied into the modeling of large-scale facilities where the blocking phenomenon can be or cannot be controlled. They also presented an approximation technique based on the expansion method to incorporate the $M/G/C/C$ queueing models into series, merge and splitting topologies of production lines. Jain and MacGregor Smith (1994) presented an analytical technique to calculate system performance measures of $M/M/C/K$ queueing networks. They analyzed series, merge and splitting topologies and in addition they explored the optimal order of the $M/M/C/K$ servers in such systems.

In Diamantidis, Papadopoulos and Heavey (2006), a flow line with parallel machines at each work-station is analyzed via the decomposition method which was presented in Section 2.5.2. The proposed approach differs from those of Ancelin and Semery (1987), Burman (1995), Jeong and Kim (1999) and Patchong and Willaeyts (2001), in that each parallel-machine work-station is not replaced by an equivalent work-station. That is, the decomposition approach is applied directly to each one of the parallel machines for each work-station without using replacement techniques.

It is expected that this direct approach will provide more accurate results than do the replacement techniques.

Regarding the non-linear flow lines, the material in the text is based on the paper by Diamantidis, Papadopoulos and Vidalis (2004). An excellent exposition of this area is given in the book by Helber (1999) in which various non-linear flow models are analyzed. Models using continuous variables are given by Tan (2001) and by Helber and Mehrrens (2003), in which times to failure and repair are exponentially distributed. Other relevant papers include Gershwin (1991), Jeong and Kim (1998), Yu and Bricker (1993), Ammar and Gershwin (1989), Dallery, Liu and Towsley (1994), Di Mascolo et al. (1991), Frein et al. (1996), Helber (1998), among others.

References

1. Alkaff, A. and Muth, E.J. (1987), The throughput rate of multistation production lines with stochastic servers, *Probability in the Engineering and Informational Sciences*, Vol. 1, pp. 309–326.
2. Altioik, T. (1982), Approximate analysis of exponential tandem queues with blocking, *European Journal of Operational Research*, Vol. 11, pp. 390–398.
3. Altioik, T. (1989), Approximate analysis of queues in series with phase-type service times and blocking, *Operations Research*, Vol. 37, pp. 601–610.
4. Altioik, T. (1997), *Performance Analysis of Manufacturing Systems*, Springer.
5. Altioik, T. and Ranjan, R. (1989), Analysis of production lines with general service times and finite buffers: A two-node decomposition approach, *Engineering Costs and Production Economics*, Vol. 17, pp. 155–165.
6. Altioik, T. and Melamed, B. (2001), *Simulation Modeling and Analysis with Arena*, Cyber Research, Inc. and Enterprise Technology Solutions, Inc.
7. Ammar, M. and Gershwin, S.B. (1989), Equivalence relations in queueing models of fork/join queueing networks with blocking, *Performance Evaluation*, Vol. 10, pp. 233–245.
8. Ancelin, B. and Semery, A. (1987), Calcul de la productivite d'une ligne integree de fabrication: CALIF, une methode analytique industrielle, *RAIRO, APII*, Vol. 21, No. 3, pp. 209–238.
9. Anderson, D.R. and Moodie, C.L. (1969), Optimal buffer storage capacity in production line systems, *International Journal of Production Research*, Vol. 7, pp. 233–240.
10. Banks, J., Carson, J.S., and Nelson, B.L. (1999), *Discrete-Event System Simulation*, 2nd Edition, Prentice Hall.
11. Benson, D. (1996), *Simulation Modeling and Optimization using PROMODEL*, PROMODEL Corporation, Orem, Utah.
12. Blumenfeld, D.E. (1990), A simple formula for estimating throughput of serial production lines with variable processing times and limited buffer capacity, *International Journal of Production Research*, Vol. 28, No. 6, pp. 1163–1182.
13. Boxma, O.J. and Konheim, A.G. (1981), Approximate analysis of exponential queueing systems with blocking, *Acta Informatica*, Vol. 15, pp. 19–66.
14. Brandwajn, A. and Jow, Y.L. (1988), An approximation method for tandem queues with blocking, *Operations Research*, Vol. 36, pp. 73–83.
15. Brateley, P., Fox, B.L., and Schrage, L.E. (1987), *A Guide to Simulation*, Springer-Verlag.

16. Burman, M.H. (1995), *New Results in Flow Line Analysis*, Ph.D. Thesis, Massachusetts Institute of Technology (M.I.T.).
17. Buzacott, J.A. (1972), The effect of station breakdowns and random processing times on the capacity of flow lines with in-process storage, *AIIE Transactions*, Vol. 4, No. 4, pp. 308–313.
18. Buzacott, J.A. and Kostelski, D. (1987), Matrix-geometric and recursive algorithm solution of a two-stage unreliable flow line, *IIE Transactions*, Vol. 19, pp. 429–438.
19. Buzacott, J.A. and Shanthikumar, J.G. (1993), *Stochastic Models of Manufacturing Systems*, Prentice Hall.
20. Caseau, P. and Pujolle, G. (1979), Throughput capacity of a sequence of transfer lines with blocking due to finite waiting room, *IEEE Transactions Software Eng.*, SE-5, pp. 631–642.
21. Cheah, J.Y. and MacGregor Smith, J. (1994), Generalized $M/G/C/C$ state dependent queueing models and pedestrian traffic flows, *Queueing Systems*, Vol. 15, pp. 365–386.
22. Colledani, M., Matta, A., and Tolio, T. (2003), Performance evaluation of production lines with finite buffer capacity producing two different products, *In Proceedings of the Fourth Aegean International Conference on Analysis of Manufacturing Systems*, pp. 231–240, Samos Island, Greece.
23. Dallery, Y., David, R., and Xie, X.L. (1988), An efficient algorithm for analysis of transfer lines with unreliable machines and finite buffers, *IIE Transactions*, Vol. 20, pp. 280–283.
24. Dallery, Y. and Frein, Y. (1993), On decomposition methods for tandem queueing networks with blocking, *Operations Research*, Vol. 41, No. 2, pp. 386–399.
25. Dallery, Y. and Gershwin, S.B. (1992), Manufacturing flow line systems: A review of models and analytical results, *Queueing Systems Theory and Applications*, Vol. 12, pp. 3–94.
26. Dallery, Y., Liu, Z., and Towsley, D. (1994), Equivalence, reversibility, symmetry and concavity properties in fork-join queueing networks with blocking, *Journal of the Association for Computing Machinery*, Vol. 41, No. 5, pp. 903–942.
27. Diamantidis, A.C., Papadopoulos, C.T., and Heavey, C. (2006), Approximate analysis of serial flow lines with multiple parallel-machine stations, *IIE Transactions*, Vol. 39, No. 4, pp. 361–375.
28. Diamantidis, A.C., Papadopoulos, C.T., and Vidalis, M.I. (2004), Exact analysis of a discrete material three station one buffer merge system with unreliable machines, *International Journal of Production Research*, Vol. 42, No. 4, pp. 651–675.
29. Di Mascolo, M., David, R., and Dallery, Y. (1991), Modeling and analysis of assembly systems with unreliable machines and finite buffers, *IIE Transactions*, Vol. 23, No. 4, pp. 315–330.
30. Dubois, D. and Forestier, J.P. (1982), Productivité et en-cours moyens d'un ensemble de deux machines séparées par une zone de stockage, *RAIRO Automatique*, Vol. 16, No. 2, pp. 105–132.
31. Fishman, G.S. (1973), *Concepts and Methods in Discrete Event Simulation*, Wiley.
32. Forestier, J.P. (1980), Modélisation stochastique et comportement asymptotique d'un système automatisé de production, *RAIRO Automatique*, Vol. 14, No. 2, pp. 127–143.
33. Freeman, D.R. (1968), A general line balancing model, *Proceedings of the 19th Annual Conference, AIIE*, Tampa, Florida, Norcross, Georgia: American Institute of Industrial Engineers, pp. 230–235.
34. Frein, Y., Commault, C., and Dallery, Y. (1996), Modeling and analysis of closed-loop production lines with unreliable machines and finite buffers, *IIE Transactions*, Vol. 28, pp. 545–554.

35. Friedman, H.D. (1965), Reduction methods for tandem queueing systems, *Operations Research*, Vol. 13, pp. 121–131.
36. Gershwin, S.B. (1987), An efficient decomposition method for the approximate evaluation of tandem queues with finite storage space and blocking, *Operations Research*, Vol. 35, pp. 291–305.
37. Gershwin, S.B. (1991), Assembly/disassembly systems: An efficient decomposition algorithm for tree structured networks, *IIE Transactions*, Vol. 23, No. 4, pp. 302–314.
38. Gershwin, S.B. (1994), *Manufacturing Systems Engineering*, Prentice Hall.
39. Gershwin, S.B. and Berman, O. (1981), Analysis of transfer lines consisting of two unreliable machines with random processing times and finite storage buffers, *AIIE Transactions*, Vol. 13, No. 1, pp. 2–11.
40. Gershwin, S.B., Maggio, N., Matta, A., Tolio, T., and Werner, L. (2001), Analysis of loop networks by decomposition, In *Proceedings of the Third Aegean International Conference on Analysis and Modeling of Manufacturing Systems*, pp. 239–248, Tinos Island, Greece.
41. Gershwin, S.B. and Schick, I.C. (1983), Modeling and analysis of three-stage transfer lines with unreliable machines and finite buffers, *Operations Research*, Vol. 31, No. 2, pp. 354–380.
42. Gosavi, H.D. and MacGregor Smith, J. (1995), Asymptotic bounds of throughput in series-parallel queueing networks, *Computers & Operations Research*, Vol. 22, No. 10, pp. 1057–1073.
43. Gun, L. and Makowski, A. (1989), An approximation method for general tandem queueing systems subject to blocking, *Proc. 1st Int'l Workshop on Queueing Networks with Blocking*, Raleigh, NC.
44. Haydon, B.J. (1973), The behaviour of systems of finite queues, *Ph.D. Thesis*, The University of New South Wales, Kensington, New South Wales, Australia.
45. Heavey, C., Papadopoulos, H.T., and Browne, J. (1993), The throughput rate of multi-station unreliable production lines, *European Journal of Operational Research*, Vol. 68, pp. 69–89.
46. Helber, S. (1998), Decomposition of unreliable assembly/disassembly networks with limited buffer capacity and random processing times, *European Journal of Operational Research*, Vol. 109, No. 1, pp. 24–42.
47. Helber, S. (1999), *Performance Analysis of Flow Lines with Non-Linear Flow of Material*, In Volume 473 of *Lecture Notes in Economics and Mathematical Systems*, Springer-Verlag.
48. Helber, S. and Jusic, H. (2004), A new decomposition approach for non-cyclic continuous material flow lines with a merging flow of material, *Annals of Operations Research*, Vol. 124, No. 1–4, pp. 117–139.
49. Helber, S. and Mehrtens, N. (2003), Exact analysis of a continuous material merge system with limited buffer capacity and three stations, In S.B. Gershwin, Y. Dallery, C.T. Papadopoulos, and J. MacGregor Smith, Editors, *Analysis and Modeling of Manufacturing Systems*, pp. 85–121, Kluwer Academic Publishers.
50. Hillier, F.S. and Boling, R.W. (1967), Finite queues in series with exponential or Erlang service times – A numerical approach, *Operations Research*, Vol. 15, pp. 286–303.
51. Hillier, F.S. and So, K.C. (1989), The assignment of extra servers to stations in tandem queueing systems with small or no buffer, *Performance Evaluation*, Vol. 10, pp. 219–231.
52. Hillier, F.S. and So, K.C. (1995), On the optimal design of tandem queueing systems with finite buffers, *Queueing Systems*, Vol. 21, pp. 245–266.

53. Hillier, F.S. and So, K.C. (1996), On the simultaneous optimization of server and work allocations in production line systems with variable processing times, *Operations Research*, Vol. 44, No. 3, pp. 435–443.
54. Hunt, G.C. (1956), Sequential arrays of waiting lines, *Operations Research*, Vol. 4, pp. 674–683.
55. Iyama, T. and Ito, S. (1987), The maximum production rate for an unbalanced multi-server flow line system with finite buffer storage, *International Journal of Production Research*, Vol. 25, No. 8, pp. 1157–1170.
56. Jain, S. and Smith, J.M. (1994), Open finite queueing networks with $M/M/C/K$ parallel servers, *Computers & Operations Research*, Vol. 21, No. 3, pp. 297–317.
57. Jeong, K.-C. and Kim, Y.-D. (1998), Performance analysis of assembly/disassembly systems with unreliable machines and random processing times, *IIE Transactions*, Vol. 30, pp. 41–53.
58. Jeong, K.-C. and Kim, Y.-D. (1999), An approximation method for performance analysis of assembly/disassembly systems with parallel-machine stations, *IIE Transactions*, Vol. 31, pp. 391–394.
59. Jun, K. and Perros, H.G. (1987), An approximate analysis of open tandem queueing networks with blocking and general service times, *Computer Science*, NCSU, Raleigh, NC.
60. Kelton, W.D., Sadowski, R.P., and Sadowski, D.A. (1998), *Simulation with Arena*, McGraw-Hill.
61. Kerbache, L. (1984), Analysis of open finite queueing networks, Ph.D. thesis, Department of Industrial Engineering and Operations Research, University of Massachusetts, Amherst, MA.
62. Kerbache, L. and MacGregor Smith, J. (1987), The generalized expansion method for open finite queueing networks, *European Journal of Operational Research*, Vol. 32, pp. 448–461.
63. Khoshnevis, B. (1994), *Discrete Systems Simulation*, McGraw-Hill.
64. Kleinrock, L. (1975), *Queueing Systems, Vol. I*, Wiley.
65. Knott, A.D. (1970), The inefficiency of a series of work-stations – a simple formula, *International Journal of Production Research*, Vol. 8, pp. 109–119.
66. Kouikoglou, V.S. and Phillis, Y.A. (2001), *Hybrid Simulation Models of Production Networks*, Kluwer Academic Publishers.
67. Kuhn, H. (2003), Analysis of automated flow line systems with repair crew interference, In S.B. Gershwin, Y. Dallery, C.T. Papadopoulos, and J. MacGregor Smith, Editors, *Analysis and Modeling of Manufacturing Systems*, pp. 155–179, Kluwer Academic Publishers.
68. Labetoulle, J. and Pujolle, G. (1980), Isolation method in a network of queues, *IEEE Transact. Software Eng.*, Vol. SE-6, No. 4.
69. Law, A.M. and Kelton, W.D. (2000), *Simulation Modeling and Analysis*, McGraw-Hill.
70. Levantesi, R., Matta, A., and Tolio, T. (1999a), Exponential two machine lines with multiple failure modes and finite buffer capacity, In *Proceedings of the Second Aegean International Conference on Analysis and Modeling of Manufacturing Systems*, pp. 103–112, Tinos Island, Greece.
71. Levantesi, R., Matta, A., and Tolio, T. (1999b), A decomposition method for the performance evaluation of production lines with random processing times, multiple failure modes and finite buffer capacity, In *Proceedings of the Second Aegean International Conference on Analysis and Modeling of Manufacturing Systems*, pp. 113–122, Tinos Island, Greece.

72. Levantesi, R., Matta, A., and Tolio, T. (2003), Performance evaluation of continuous production lines with machines having different processing times and multiple failure modes, *Performance Evaluation*, Vol. 51, pp. 247–268.
73. Lim, J.-T., Meerkov, S.M., and Top, F. (1990), Homogeneous, asymptotically reliable serial production lines: Theory and a case study, *IEEE Transactions on Automatic Control*, Vol. 35, No. 5, pp. 524–534.
74. Magazine, M.J. and Stecke, K.E. (1996), Throughput for production lines with serial work-stations and parallel service facilities, *Performance Evaluation*, Vol. 25, No. 3, pp. 211–232.
75. Makino, T. (1964), On the mean passage time concerning some queueing problems of the tandem type, *J. Opns Res. Soc. Japan*, Vol. 7, pp. 17–47.
76. Muth, E.J. (1984), Stochastic processes and their network representations associated with a production line queueing model, *European Journal of Operational Research*, Vol. 15, pp. 63–83.
77. Muth, E.J. (1987), An update on analytical models of serial transfer lines, *Research Report*, No. 87-15, Gainesville, FL: Department of Industrial and Systems Engineering, University of Florida.
78. Nemec, J.E. (1999), *Diffusion and Decomposition Approximations of Stochastic Models of Multiclass Processing Networks*, Ph.D. Thesis, Massachusetts Institute of Technology (M.I.T.).
79. Neuts, M.F. (1981), *Matrix-Geometric Solutions in Stochastic Models – An Algorithmic Approach*, The Johns Hopkins University Press.
80. Onvural, R.O. and Perros, H.G. (1986), On equivalencies of blocking mechanisms in queueing networks with blocking, *Operations Research Letters*, Vol. 5, No. 6, pp. 293–298.
81. Papadopoulos, H.T. (1989), Mathematical modelling of reliable production lines, Ph.D. Thesis, University College Galway, Ireland.
82. Papadopoulos, H.T. (1996), An analytic formula for the mean throughput of K-station production lines with no intermediate buffers, *European Journal of Operational Research*, Vol. 91, pp. 481–494.
83. Papadopoulos, H.T., Heavey, C., and Browne, J. (1993), *Queueing Theory in Manufacturing Systems Analysis and Design*, Chapman & Hall.
84. Papadopoulos, H.T., Heavey, C., and O’Kelly, M.E.J. (1989) Throughput rate of multistation reliable production lines with inter-station buffers (I) Exponential Case, *Computers in Industry*, Vol. 13, No. 3, pp. 229–244.
85. Papadopoulos, H.T., Heavey, C., and O’Kelly, M.E.J. (1990), Throughput rate of multistation reliable production lines with inter station buffers: II Erlang case, *Computers in Industry*, Vol. 13, No. 4, pp. 317–335.
86. Papadopoulos, H.T. and O’Kelly, M.E.J. (1989) A recursive algorithm for generating the transition matrices of multistation series production lines, *Computers in Industry*, Vol. 12, pp. 227–240.
87. Patchong, A. and Willaey, D. (2001), Modeling and analysis of an unreliable flow line composed of parallel-machine stages, *IIE Transactions*, Vol. 33, pp. 559–568.
88. Perros, H. (1994), *Queueing Networks with Blocking - Exact and Approximate Solutions*, Oxford University Press.
89. Perros, H.G. and Aliot, T. (1986), Approximate analysis of open networks of queues with blocking, tandem configurations, *IEEE Trans. Soft. Eng.*, SE-12, pp. 450–461.
90. Pritsker, A.A.B. (1986), *Introduction to Simulation and Slam II*, 3rd Edition, Halsted.

91. Sevast'yanov, B.A. (1962) Influence of stage bin capacity on the average standstill time of a production line, *Theory of Probability and its Applications*, Vol. 7, No. 4, pp. 429–438.
92. Systems Modeling Corporation (1999), *Guide to Arena Standard Edition*, Sewickley.
93. Takahashi, Y., Miyahara, H., and Hasegawa, T. (1980), An approximation method for open restricted queueing network, *Operations Research*, Vol. 28, pp. 594–602.
94. Tan, B. (2001), A three-station merge system with unreliable stations and a shared buffer, *Mathematical and Computer Modeling*, Vol. 33, pp. 1011–1026.
95. Tempelmeier, H. and Burger, M. (2001), Performance evaluation of unbalanced flow lines with general distributed processing times, failures and imperfect production, *IIE Transactions*, Vol. 33, No. 4, pp. 293–302.
96. Tolio, T. and Matta, A. (1998), A method for performance evaluation of automated flow lines, *Annals of the CIRP*, Vol. 47, No. 1, pp. 373–376.
97. Tolio, T., Matta, A., and Gershwin, S.B. (2002), Analysis of two-machine lines with multiple failure modes, *IIE Transactions*, Vol. 34, pp. 51–62.
98. Van Dijk, N. and van der Wal, J. (1989), Simple bounds and monotonicity results for finite multi-server exponential tandem queues, *Queueing Systems Theory and Applications*, Vol. 4, pp. 1–16.
99. Vidalis, M.I. and Papadopoulos, H.T. (2001), A recursive algorithm for generating the transition matrices of multistation multiserver exponential queueing networks, *Computers & Operations Research*, Vol. 28, No. 9, pp. 853–883.
100. Yu, K.-Y.C. and Bricker, D.L. (1993), Analysis of a Markov chain model of a multistage manufacturing system with inspection, rejection, and rework, *IIE Transactions*, Vol. 25, No. 1, pp. 109–112.
101. Zimmern, B. (1956), Etudes de la propagation des arrêts aleatoires dans les chaines de production, *Review Statistical Applications*, Vol. 4, pp. 85–104.

Analysis and Design of Discrete Part Production Lines

Papadopoulos, C.T.; O'Kelly, M.E.J.; Vidalis, M.J.; Spinellis, D.

2009, XX, 279 p. 113 illus., 18 illus. in color., Hardcover

ISBN: 978-0-387-89493-5