INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

General Open and Closed Queueing Networks with Blocking: A Unified Framework for Approximation

Mark Vroblefski, R. Ramesh, Stanley Zionts,

INFORMS Journal on Computing
Vol. 12, No. 4, Fall 2000

0899-1499/100/1204-0299 $05.00
© 2000 INFORMS

General Open and Closed Queueing Networks with Blocking: A Unified Framework for Approximation

MARK VROBLEFSKI / *Department of Management Science and Information Technology, Virginia Polytechnic Institute and State University, 1007 Pamplin Hall (0235), Blacksburg, VA 24061, Email: mvro@vt.edu*

R. RAMESH and STANLEY ZIONTS / *Department of Management Science & Systems, School of Management, State University of New York at Buffalo, Buffalo, NY 14260, Email: rramesh@acsu.buffalo.edu, szionts@buffalo.edu*

(Received: March 1999; revised: March 2000; accepted: May 2000)

In this paper, we develop a unified framework for approximating open and closed queueing networks under any general blocking protocol by extending and generalizing the approximation algorithm for open tandem queues under minimal blocking presented in Di Mascolo et al. (1996). The proposed framework is based on decomposition. We develop decomposition structures and analysis algorithms for any general blocking system using the framework. The proposed algorithms have been extensively tested using simulations as a benchmarking device. The results show that the proposed framework yields robust, reliable, and accurate estimates of system characteristics, such as throughput and Work-In-Process inventory in a wide range of system configurations. The computational load is minimal. The unified framework presents a highly useful set of tools of analysis for queueing-system designers to use in evaluating the performance under numerous design alternatives. Directions for future research are presented, with a focus on critical application areas such as packet-switching-network design and cellular manufacturing.

Most production and communication systems can be modeled as queueing networks. An analysis of queueing networks provides a basis for the optimal design of several real-life systems such as assembly production lines, network-structured cellular manufacturing systems, and packet-switching telecommunication networks. Inherent in such real-life systems are several complexities that make an exact analysis of the underlying queueing models almost impossible. Some of the major complexities include the blocking phenomenon resulting from finite waiting spaces in the respective queues in a network, the network topology (open/closed), general service-time distributions, and the size of the network. The importance of these issues in practical systems design has led to considerable attention in the literature on approximation techniques for queueing-system analysis.

In this paper, we develop a unified framework for approximately solving a queueing system under blocking. To illustrate our methodology better, we focus on tandem networks and cyclic networks with limited waiting space at each service point. We develop approximation strategies for open and closed queueing systems under four types of blocking: *manufacturing*, *communication*, *minimal*, and *general*. The proposed algorithms in these cases are based on a unified model of serial queueing systems. We assume general service-time distributions at the server nodes and both saturated input supply and output demand. The saturation assumption is intended to simplify our development, and the proposed framework can easily be modified to handle unsaturated supply and demand (Di Mascolo et al. 1996). The framework is unified for a general class of blocking protocols for the following reasons: (i) a common set of modeling tools such as the kanban characterization of queueing systems, synchronization stations that link the flows from a production cell to the next, closed queueing analysis of circulating kanbans within a cell, and open system analysis of the production stage queues in the cells are used in each blocking protocol; (ii) a common strategy for cell-level analysis and a common iterative algorithm for approximating over the cells in a serial line are used; and (iii) the same approach is used for approximating both open and closed queueing systems. However, the individual protocols differ in how these models are formulated; accordingly, the specific models of analysis and approximation within each cell are unique to each protocol, although developed from the same set of modeling tools.

Existing work can be classified as: Blocked Open Queueing Networks (BOQN) and Blocked Closed Queueing Networks (BCQN). The existing literature addresses primarily BOQN systems, and considers mainly manufacturing and communication blocking (Perros 1994). Studies on BCQN systems are much fewer in number because of the additional complexity arising out of a fixed number of pallets circulating in such systems (Onvural 1990). The work of Di Mascolo et al. (1996) is perhaps a first approximation scheme for BOQN systems under minimal blocking. General blocking, which encompasses all types of blocking was introduced by Cheng and Yao (1994). To our knowledge, there exists no approximation algorithm for analyzing BOQN or BCQN systems under general blocking. However, general blocking forms the basis of a large set of queueing-system design configurations that could significantly outperform designs under the other blocking schemes (Ramesh et al. 1997a, 1997b). Based on these considerations, there is a need for a

Subject classifications: Queues-approximations, queues-tandem, queues-algorithms
Other key words: Inventory-production: approximations-heuristics

unified and efficient approximation framework for both open and closed queueing networks under *all* types of blocking. This need motivated our research.

Before we present our approach, we review relevant literature. An exact analysis of BOQN systems with more than two servers is difficult, so approximations and simulations are widely used. Several approximation schemes are based on decompositions of queueing networks into subsystems that are analyzed separately. Iterative procedures that combine the subsystem analyses into overall network analyses constitute the core of such approximation strategies. Some important works of this type include Dallery (1990), Dallery and Frein (1993), Di Mascolo et al. (1996), and Perros et al. (1988). Perros (1994) presents a detailed classification of approximation schemes for BOQN systems. Onvural (1990) presents a survey of exact and approximate methods for BCQN systems. While two server BCQN systems have been well analyzed (Akyildiz 1987, Gordon and Newell 1967a, 1967b, Van Dijk and Tijms 1986), studies on systems with more than two servers are rather few (see e.g. Onvural 1990). In the following discussion, we address the past research that constitutes the building blocks of the proposed framework and develop its underlying rationale.

Di Mascolo et al. (1996) developed an approximation scheme for a kanban-controlled open production line under minimal blocking by decomposing the line into production stages, analyzing the stages independently, and then linking the analyses into an overall iterative approximation procedure. Each stage is modeled as a Nonblocking Closed Queueing Network (NBCQN) with a fixed number of kanbans circulating within a production stage constituting the loop. The authors introduce the notion of a *synchronization station* between adjoining stages, and analyze each NBCQN as a subsystem constituting a production stage and two adjoining synchronization stations, one on either side of the stage. The synchronization stations yield closed-form Markov solutions under minimal blocking, and the production stage is analyzed as a $\lambda(n)/C_2/1/N$ queue using the method of Marie (1980). In the analysis of Marie (1980), the departure rates from a server are state-dependent, although the service time follows a fixed phase-type (coxian) distribution. This is due to the fact that the arrival rates to the queue in front of the server are state-dependent. Furthermore, as long as the distributions of the service times are phase-type (combination of exponential distributions), the system can be modeled as a continuous-time Markov chain with a product form solution (Gordon and Newell 1967b, Di Mascolo et al. 1996). Therefore, by integrating the three analyses together with a product-form network solution for NBCQN systems, an iterative procedure for the overall analysis of the subsystem is developed. Details and references on these product-form methods can be found in Baynat and Dallery (1993), Bruell and Balbo (1980), and Baskett et al. (1975). Finally, linking the subsystem analyses in sequential downstream and upstream orders, a successive-iteration scheme for the approximation of the entire line is obtained.

The proposed unified framework for BOQN and BCQN systems builds on and generalizes the above decomposition strategy. Queues in tandem under each blocking protocol have an equivalent kanban system configuration (Mitra and Mitrani 1990, 1991, Cheng and Yao 1994). Consequently, we adopt this equivalence and model BOQN and BCQN systems under various blocking protocols using their kanban system counterparts. First, we generalize the decomposition strategy of Di Mascolo et al. (1996) to approximate kanban systems under various blocking protocols, yielding a framework for the analysis of BOQN systems in general. Next, we extend this framework to BCQN systems by embedding an analysis of the corresponding BOQN system within an NBCQN analysis of the pallets in circulation. The strategy in this case is to realize BOQN flow characteristics equivalent to those of the BCQN. As a result, we obtain a unified framework for analysis of BOQN as well as BCQN systems under any blocking mechanism.

Extensive empirical evaluations of the proposed framework to assess the computational effort and the quality of the approximations have been carried out. The quality of the approximations has been measured in terms of two parameters: system throughput (ST) and the total work in process (WIP) inventory. The approximations have been evaluated in comparison with measures obtained through simulation. The results show that the proposed framework yields reliable approximations with minimal computational effort for a significant class of queueing networks. Furthermore, the results are especially significant as the approximations work with any general service-time distribution, whereas some knowledge of the service distribution is required for simulation.

The organization of the paper is as follows. Section 1 presents the approximation framework for BOQN systems. Section 2 develops the framework to approximate BCQN systems and concludes with the unified model. Section 3 presents the computational results, and Section 4 presents the conclusions and directions for future research.

1. Blocked Open Queueing Networks

We first introduce some basic notation in the following analysis of BOQN systems. Let $i = 1, \dots, N$ denote a sequence of servers operating in series with finite buffer space at each service cell. The input to the system (raw material) enters cell 1 and the output (finished products) leaves cell N. We assume saturated input and output conditions in the queueing system and general service-time distributions at the cells. Let k_i denote the buffer space available at service cell i, for $i = 1, \dots, N$. We develop the different blocking conditions in this queueing system as follows.

The parts arriving at a cell form a waiting line in the buffer area. Assume that each cell has a *roving* server, who moves from buffer to buffer to complete the processing required. A completed part is immediately transferred to the next cell if a buffer is available and the blocking protocol admits the transfer. Otherwise, the finished part is retained in its existing buffer and the roving server is free to move on to the next buffer in that cell. In this operational scheme, we define two more control parameters for each cell as follows. Let a_i and b_i denote the maximum number of parts at cell i

that are awaiting service and awaiting outward transfer after service completion, respectively. It follows that, $a_i \leqslant k_i$, $b_i \leqslant k_i$ and $a_i + b_i \geqslant k_i$, $i = 2, \ldots, N-1$, $a_1 = k_1 = b_1$ (due to supply saturation), $a_N = k_N$, $b_N = 0$ (due to demand saturation), and $k_i > 0 \ \forall \ i$. These conditions define the *general blocking* control structure in tandem queues (Cheng and Yao 1994). The other well-known blocking structures are obtained from the general blocking defining conditions as follows: $\{a_i = k_i, b_i = 1$ for $i = 1, \ldots, N-1, a_N = k_N, b_N = 0\}$ defines *manufacturing blocking*, $\{a_i = b_i = k_i$, for $i = 1, \ldots, N-1, a_N = k_N, b_N = 0\}$ defines *minimal blocking*, and the same conditions of manufacturing blocking modified with $b_i = 0$ for $i = 1, \ldots, N-1$ define *communication blocking* (Ramesh et al. 1997a, Buzacott 1988, Zipkin 1989).

In general, let $(\mathbf{a}, \mathbf{b}, \mathbf{k}) = (a_i, b_i, k_i)_{i=1}^{i=N}$ define the control parameters of the general blocking tandem queue system. We denote the sets of control parameters for manufacturing, minimal, and communication blocking protocols as **(k, 1, k)**, **(k, k, k)**, and **(k, 0, k)**, respectively. Furthermore, it has been shown that any general blocking system **(a, b, k)** has equivalent systems $(\tilde{\mathbf{a}}, \tilde{\mathbf{k}}, \tilde{\mathbf{k}})$ and $(\hat{\mathbf{k}}, \hat{\mathbf{b}}, \hat{\mathbf{k}})$ that yield identical time epochs in the flow of parts throughout the line (Ramesh et al. 1997a, 1997b). These equivalent configurations reduce general blocking systems to just two sets of parameters, and are termed *selective input* and *selective output* control systems, respectively. We address general blocking systems using their selective output control equivalents. The above control parameters also have specific functionalities in the kanban equivalents of queues in tandem as follows. The parameters a_i, b_i, and k_i denote the number of input buffers, output buffers and the circulating kanban cards at each cell i (Mitra and Mitrani 1990). The proposed approximation framework employs the kanban equivalents, and includes algorithms for **(k, b, k)**, **(k, 1, k)** and **(k, 0, k)** blocking configurations.

In the kanban version of queues in tandem analysis, Di Mascolo et al. (1996) introduce the concept of *synchronization stations*. A synchronization station is a *virtual* station positioned between adjoining service points, and is intended to capture the coordination between the two cells in the transfer of parts from one to the other. A synchronization station between cells i and $(i+1)$ consists of two buffer areas: upstream and downstream. The upstream area corresponds to the output buffers of cell i and the downstream area corresponds to the bulletin board to which the released kanbans return in cell $(i+1)$. A finished part in the upstream area is matched with a kanban in the downstream area, and together are taken to the input queue awaiting service at cell $(i+1)$. In the decomposition strategy, the entire line is broken into subsystems, with each subsystem consisting of the synchronization station preceding a service point, the input queue at the service point, and the following synchronization station. Analyzing each subsystem independently, the entire line is approximated using a forward-backward iterative passing scheme among the subsystems (Di Mascolo 1996).

We employ the above scheme in the proposed generalized framework. We develop the structure of the synchronization stations, their methods of analysis, and the iterative schemes of line approximation under different blocking protocols. We first present the framework for approximation, and subsequently specialize it for different protocols.

1.1 BOQN Approximation Framework

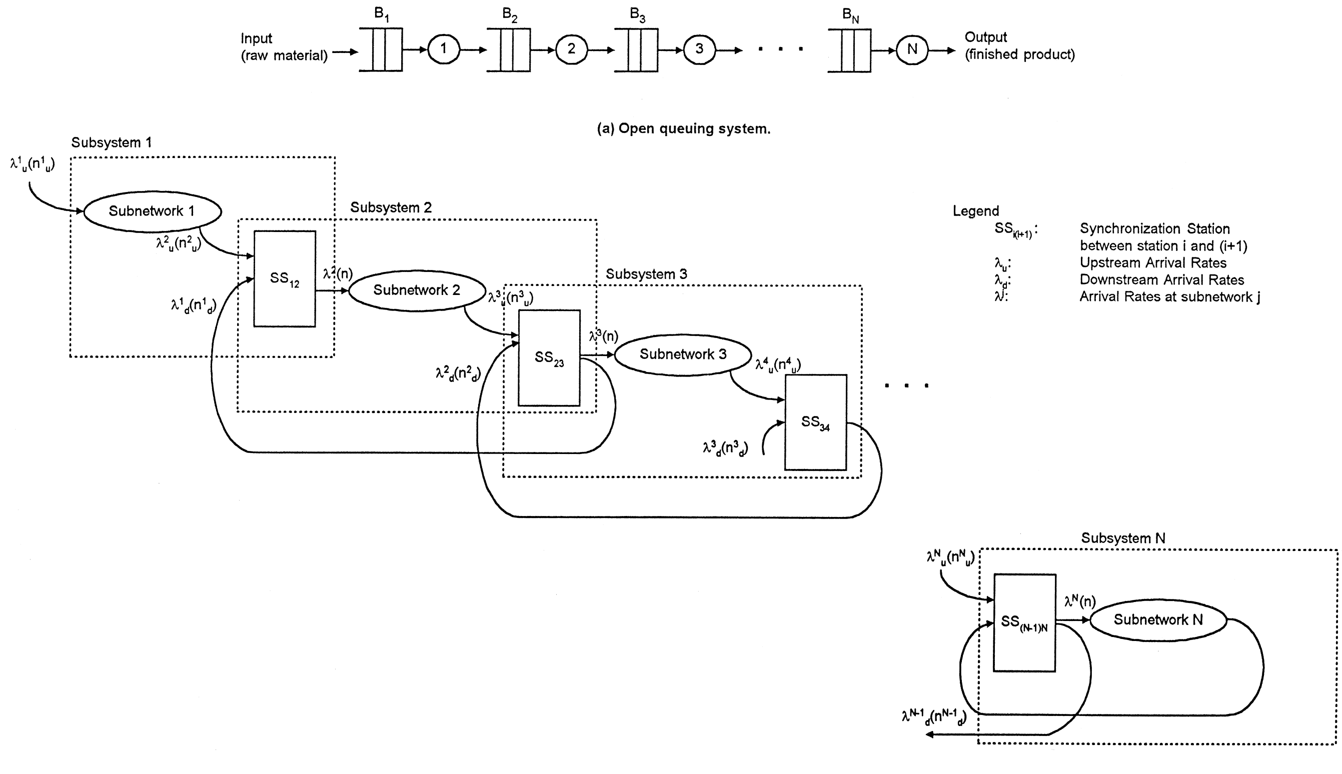
The BOQN approximation framework is based on the decomposition strategy of DiMascolo et al. (1996). Consider the open tandem queueing system shown in Figure 1(a). Using the kanban version of this line, the system is decomposed into a sequence of subsystems as shown in Figure 1(b). A synchronization station $\mathrm{SS}_{i(i+1)}$ participates in the individual analyses of subsystems i and $(i+1)$. The queueing system at each service cell in Figure 1(a) is indicated as the subnetwork in Figure 1(b). The upstream arrivals correspond to the flow of parts through the line, and the downstream arrivals pertain to the returning kanbans at each cell. The arrival rates are state-dependent, and are denoted as follows. The upstream arrival rates at subsystem i are $\lambda^i{}_u(n^i{}_u)$, $0 \leqslant n^i{}_u \leqslant k_{i-1}$, where $n^i{}_u$ is the number of parts awaiting removal at the output buffer area of subsystem i. The downstream arrival rates at subsystem i are $\lambda^i{}_d(n^i{}_d)$, $0 \leqslant n^i{}_d \leqslant k_{i+1}$, where $n^i{}_d$ is the number of kanbans already at the bulletin board of subsystem $(i+1)$. The output buffer area of cell i and the bulletin board of cell $(i+1)$ together constitute the synchronization station $\mathrm{SS}_{i(i+1)}$. Let $\mu^i(n)$, $1 \leqslant n \leqslant k_i$ denote the state dependent service rates at cell i. Let $\lambda^j(n)$, $0 \leqslant n \leqslant k_i$ denote the state-dependent arrival rates of the combination of parts and kanbans at subnetwork j. For the sake of simplicity, we assume saturated input supply and output demand in the tandem queueing system in this presentation. This is not a limitation of the proposed framework, and unsaturated supply and demand can easily be incorporated by adding a synchronization station at the beginning of the line to feed inputs according to a saturated supply and another synchronization station at the very end to pull the flow with a saturated demand (see Di Mascolo et al. 1996 for details). The structures of these synchronization stations are independent of the blocking protocols that may be used within the queueing system. We now present the overall approximation framework as follows. Subsequently, we specialize this framework to specific blocking protocols.

Framework: BOQN

Step 0: {Initialization}
Fix all downstream arrival rate parameters λ_d's to some initial values. The suggested initial values could be the mean service rates at the corresponding subsystems.

Step 1: {Forward Pass}
Starting from subsystem 1, successively solve for each subsystem till subsystem $(N-1)$; in this analysis, solve for the parameters $\lambda_u^{(i+1)}$ given the parameters λ_u^i and λ_d^i while analyzing subsystem i.

Step 2: {Backward Pass}
Starting from subsystem N, successively solve for each subsystem till subsystem 2; in this analysis, solve for the parameters $\lambda_d^{(i-1)}$ given the parameters $\lambda_u^{(i-1)}$ and λ_d^i while analyzing subsystem i.



(a) Open queuing system.

(b) Decomposition into subsystems.

Figure 1. The decomposition framework.

Step 3: {Convergence Test}
Check for convergence of the λ_d and λ_u parameters at all the subsystems. If convergence is reached according to the criteria used, then stop; else return to Step 1.

The above framework involves the independent analyses of the subsystems in steps 1 and 2. Modeling each subsystem as an NBCQN with the circulating kanbans, the overall strategy of these analyses is presented as follows.

Algorithm: Subsystem_Analysis

Step 0: {Initialization}
Consider subsystem i. Assume the unknown state-dependent service rates (μ) at each station of subsystem i ($\mathrm{SS}_{(i-1)i}$, subnetwork i, $\mathrm{SS}_{i(i+1)}$). Initially, all values of μ at each station can be set at the mean service rate of the subsystem (see Di Mascolo et al. 1996 for a discussion on setting initial values). Note that these values will be recomputed until stability is reached in the following steps.

Step 1: {NBCQN Modeling}
Modeling the circulation of kanbans within subsystem i as an NBCQN, determine the state-dependent arrival rates (λ) at each station using the algorithm of Buzen (1973) (also see Bruell and Balbo 1980).

Step 2: {Throughput Determination}
For each station in subsystem i, determine the state-dependent throughput rates (v), given the arrival rates (λ). In this case, employ Markov analyses for synchronization stations and the algorithm of Marie (1980) for the subnetwork. The individual stationwise analysis requires the underlying subsystem be modeled as an NBCQN.

Step 3: {Service Rate Determination}
Set the service rates (μ) of each station to their corresponding throughput rates (v).

Step 4: {Convergence Test}
Check for convergence of the μ parameters at each station of the subsystem. If convergence is reached according to the criteria used, then stop; else return to Step 1.

The key to the above analysis lies in the ability to model each subsystem as an NBCQN under a variety of blocking protocols. The requirements of NBCQN modeling can be summarized as follows:

(a) When a part and a kanban are found at a synchronization station, the protocol should permit an immediate transfer of them together to the following subnetwork. This implies no blocking at the synchronization stations.

(b) When a service is completed at a subnetwork, the part should be transferred immediately to the next waiting

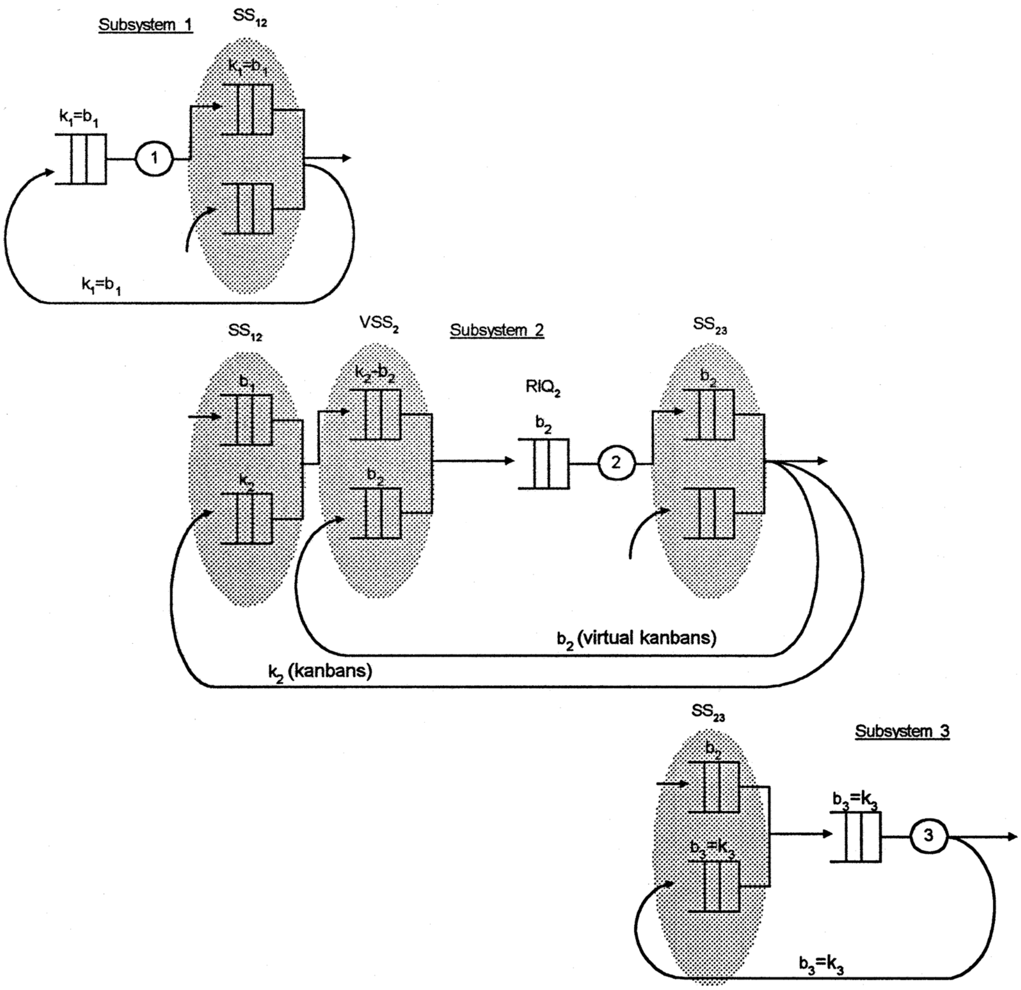


Figure 2. General blocking decomposition.

station (which could be either within the subnetwork or at the following synchronization station). This implies no blocking at the subnetwork.

In the following sections, we develop architectures of subsystems that satisfy these requirements under all types of blocking protocols.

1.2 General Blocking Systems

Without loss of generality, the decomposition strategy under general blocking is illustrated in Figure 2 using three stages. Extensions to more than three stages is straightforward. In this Figure, the line is decomposed into three subsystems, corresponding to the three service cells. Subsystems 1 and 3 are special in the sense that they mark the beginning and end of the line, respectively. The decomposition structure of subsystem 2 will be repeated at other intermediate subsystems if the line has more than three stages. Each subsystem is modeled as a NBCQN as follows.

First consider subsystem 2, which represents any intermediate stage in the line. The subnetwork component of this subsystem is buffered between the synchronization stations SS$_{12}$ and SS$_{23}$. This subsystem is a closed queueing network with the kanbans of cell 2 circulating within the subsystem. This is shown in the circulation returning from SS$_{23}$ to SS$_{12}$ in Figure 2. Note that the bulletin board (downstream component) of a synchronization station SS$_{23}$ has a capacity to accommodate up to k_3 kanbans, and its output buffer area (upstream component) has b_2 spaces. Now consider SS$_{12}$. When a part and a kanban are matched at SS$_{12}$, they together move into the input buffer area of cell 2. Since we employ the selective output control equivalent of general blocking, we have $a_i = k_i \; \forall i$, and hence, there will always be a space in the input buffer area when this match occurs. However, it is necessary to ensure that the roving server does not begin processing on an input part if the output buffers in the cell are already full. Note that this situation can arise since $b_2 < k_2$ in the selective output control protocol. Consequently, blocking could arise within the subsystem. Therefore, in order to model the subsystem as an NBCQN, we develop the following strategy.

First, we classify the parts entering the input area with kanbans into two sets: those for which a space in the output

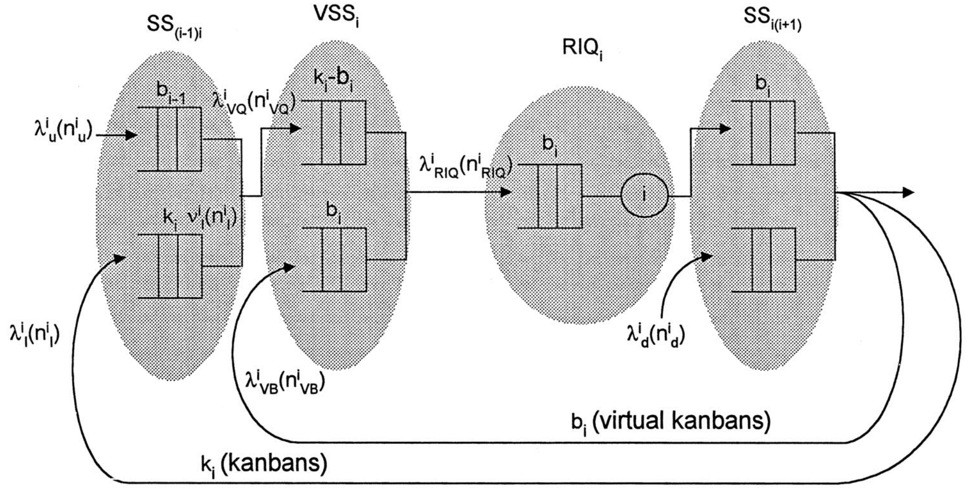


Figure 3. Analysis of subsystem $1 < i < N$ under general blocking.

area is available for transfer after service completion, and those for which no such space is available. These sets are denoted as A and U, respectively. Let $x_2 \leqslant b_2$ denote the number of finished parts awaiting removal in the output buffer area of SS_{23}. Therefore, if $|U| > 0$, then $|A| + x_2 = b_2$, and if $|A| + x_2 < b_2$, then $|U| = 0$. Using these conditions, we structurally segregate the sets A and U as follows. We introduce another type of kanban, termed the *virtual kanbans* within each subsystem. The cell 2 is assigned b_2 virtual kanbans, and are posted at a *virtual bulletin board*, which is positioned between the main bulletin board and the service station. When a part and a kanban leave synchronization station S_{12}, they first enter a queue called the *virtual queue*. The virtual queue and the virtual bulletin board together constitute a *virtual synchronization station*, and operate the same way as the other synchronization stations. A part with a kanban at the virtual queue is matched with a virtual kanban at the virtual bulletin board, and together are taken to the *real input queue*. The server is allowed to process only those parts from the real input queue. Upon service completion, the part and the two kanbans are moved to the output buffer area in SS_{23}. When a kanban from cell 3 arrives at the bulletin board of SS_{23}, the part is detached from the two current kanbans, matched with a kanban from cell 3, and moved to the virtual queue of subsystem 3. Finally, the detached kanbans are returned to their respective bulletin boards in subsystem 2.

The above design, illustrated using subsystem 2, is formalized for any intermediate subsystem $1 < i < N$ in Figure 3. Each cell i has k_i kanbans, $a_i = k_i$ input buffers and $b_i < k_i$ output buffers. The subsystem modeling cell i is composed of the following sequence of stations: synchronization station $SS_{(i-1)i}$, a virtual synchronization station denoted as VSS_i, a real input queue denoted as RIQ_i, and a synchronization station $SS_{i(i+1)}$. The output buffer area and the bulletin board of a synchronization station $SS_{i(i+1)}$ have b_i and k_{i+1} waiting spaces, respectively. The virtual queue and the virtual bulletin board at VSS_i have $(k_i - b_i)$ and b_i waiting spaces, respectively. The real input queue RIQ_i has b_i waiting spaces.

As can be seen from Figures 2 and 3, a subsystem $1 < i < N$ involves two *nested* CQNs. The kanbans among the stations $SS_{(i-1)i}$, VSS_i, RIQ_i, and $SS_{i(i+1)}$ constitute the *real* CQN, while the virtual kanbans circulating among VSS_i, RIQ_i, and $SS_{i(i+1)}$ constitute a *virtual* CQN. The following theorem establishes the correctness of modeling cell i under general blocking using the nested CQN structure.

Theorem 1. *The nested CQN structure in subsystem $1 < i < N$ yields the same time epochs and flow control as in general blocking. Furthermore, each CQN in the nested scheme is nonblocking.*

Proof: Consider $SS_{(i-1)i}$. The capacities of the output area and the bulletin board in $SS_{(i-1)i}$ are as specified in the general blocking protocol. When a part and a kanban are matched at $SS_{(i-1)i}$, they are immediately transferred to the virtual queue in VSS_i without blocking. This is because: (i) the virtual queue has a capacity of $k_i - b_i$, (ii) when the virtual queue is non empty, the virtual bulletin board should be empty (otherwise a match would occur at VSS_i), and hence, (iii) when the virtual queue is full, there can be no kanbans at the bulletin board of $SS_{(i-1)i}$. This can be observed from the circulations in the two CQNs. Next, there can be no blocking in the virtual CQN as there are exactly b_i spaces in each station of this network. Finally, since the flows in the two CQNs are identical between VSS_i, RIQ_i, and $SS_{i(i+1)}$, both the CQNs are nonblocked.

The nonblocked transfers out of $SS_{(i-1)i}$ are the same time epochs as in the underlying general blocking protocol. The admitted parts are simply partitioned into two sets U and A, with U waiting at the virtual queue and A at the real queue. Since the CQNs are nonblocking and the virtual CQN ensures that there can be at most b_i parts between the real queue and the output buffer area of cell i, the time epochs in the virtual CQN correspond to those in the underlying protocol. ■

The above virtual system modeling is not needed for cells 1 and N, as they are basically nonblocking networks themselves (see Figure 2). Consequently, subsystems 1 and N can be analyzed using the procedure given in Di Mascolo et al. (1996). We develop an algorithm to analyze a subsystem $1 < i < N$ using the framework presented earlier, as follows.

First, consider $\mathrm{SS}_{(i-1)i}$. The state-dependent arrival rates of the returning kanbans at $\mathrm{SS}_{(i-1)i}$ are denoted as $\lambda_{\mathrm{I}}^i(n_{\mathrm{I}}^i)$, $n_{\mathrm{I}}^i = 0, \ldots, \mathrm{k_i}$, where n_{I}^i is the number of kanbans already present at $\mathrm{SS}_{(i-1)i}$. Similarly, let the state-dependent arrival rates of parts into $\mathrm{SS}_{(i-1)i}$ be denoted as $\lambda_u^i(n_u^i)$, $n_u^i = 0, \ldots, b_{i-1}$, where n_u^i is the number of parts already at $\mathrm{SS}_{(i-1)i}$. The station $\mathrm{SS}_{(i-1)i}$ can be independently modeled as a Markov chain fed by two Markovian processes, and yields a closed form solution (Di Mascolo et al. 1996). The Markov analysis of $\mathrm{SS}_{(i-1)i}$ yields the state-dependent throughput rates from this station denoted as $v_{\mathrm{I}}^i(n_{\mathrm{I}}^i)$, $n_{\mathrm{I}}^i = 0, \ldots, k_i$. The combination of a part and kanban from $\mathrm{SS}_{(i-1)i}$ enter the virtual queue, whose capacity is $(k_i - b_i)$. Let $\lambda_{\mathrm{VQ}}^i(n_{\mathrm{VQ}}^i)$, $n_{\mathrm{VQ}}^i = 0, \ldots, (k_i - b_i)$ denote the state-dependent arrival rates at the virtual queue where n_{VQ}^i is the number of parts already in this queue. The following Lemma establishes a mapping from n_{I}^i to n_{VQ}^i.

Lemma 1. $n_{\mathrm{VQ}}^i = \max\{0, k_i - b_i - n_{\mathrm{I}}^i\}$, $1 < i < \mathrm{N}$

Proof. Note that n_{I}^i can be any number in the discrete interval between 0 and k_i. First, consider the case where $k_i - b_i - n_{\mathrm{I}}^i \leqslant 0$. If n_{I}^i kanbans are at the bulletin board of $\mathrm{SS}_{(i-1)i}$, then $k_i - n_{\mathrm{I}}^i$ kanbans, along with their accompanying parts, can be found anywhere in the virtual CQN. However, since b_i virtual kanbans are available and $k_i - n_{\mathrm{I}}^i \leqslant \mathrm{b_i}$, clearly the $k_i - n_{\mathrm{I}}^i$ kanbans should exist between RIQ_i and the output area. Hence, $n_{\mathrm{VQ}}^i = 0$ in this case. Now, consider the case when $k_i - b_i - n_{\mathrm{I}}^i > 0$. Since $k_i - n_{\mathrm{I}}^i > b_i$, all virtual kanbans are in use between RIQ_i and the output area. Consequently, $n_{\mathrm{VQ}}^i = k_i - b_i - n_1^i$. ■

The above Lemma is used to match the throughput rates from $\mathrm{SS}_{(i-1)i}$ with the arrival rates at the virtual queue in an iterative approximation scheme for the nested CQNs. Since n_{VQ}^i is *uniquely* mapped onto n_{I}^i when $n_{\mathrm{I}}^i < k_i - b_i$, we have

$$n_{\mathrm{VQ}}^i = k_i - b_i - n_{\mathrm{I}}^i \tag{1}$$

$$\lambda_{\mathrm{VQ}}^i(n_{\mathrm{VQ}}^i) = v_{\mathrm{I}}^i(n_{\mathrm{I}}^i)\ \ \forall n_{\mathrm{I}}^i \leqslant k_i - b_i. \tag{2}$$

However, $n_{\mathrm{VQ}}^i = 0\ \forall n_{\mathrm{I}}^i \geqslant k_i - b_i$. Consequently, we approximate $\lambda_{\mathrm{VQ}}^i(n_{\mathrm{VQ}}^i)$ in this case as follows.

$$\lambda_{\mathrm{VQ}}^i(0) = \left\{ \sum_{\eta=k_i-b_i}^{k_i} p_{\mathrm{I}}^i(\eta) v_{\mathrm{I}}^i(\eta) \right\} \Bigg/ \sum_{\eta=k_i-b_i}^{k_i} p_{\mathrm{I}}^i(\eta) \tag{3}$$

where $p_{\mathrm{I}}^i(\eta)$ is the probability of η kanbans at the bulletin board of $\mathrm{SS}_{(i-1)i}$. This approximation is a weighted average of the throughput rates $v_{\mathrm{I}}^i(\eta)$, $\eta \geqslant k_i - b_i$, weighted by the steady state probabilities $p_{\mathrm{I}}^i(\eta)$ derived from the Markov analysis of $\mathrm{SS}_{(i-1)i}$.

Now, consider VSS_i. The state-dependent arrival rates of the returning virtual kanbans at VSS_i are denoted as $\lambda_{\mathrm{VB}}^i(n_{\mathrm{VB}}^i)$, $n_{\mathrm{VB}}^i = 0, \ldots, b_i$, where n_{VB}^i is the number of virtual kanbans already present at the virtual bulletin board. The following Lemma establishes a mapping from n_{I}^i to n_{VB}^i.

Lemma 2. $n_{\mathrm{VB}}^i = \max\{0, n_{\mathrm{I}}^i - k_i + b_i\}$, $1 < i < N$.

Proof: We analyze again in terms of two cases: (i) $n_{\mathrm{I}}^i \geqslant k_i - b_i$ and (ii) $n_{\mathrm{I}}^i < k_i - b_i$. First, consider case (i). As in Lemma 1, since $n_{\mathrm{VQ}}^i = 0$ and $k_i - n_{\mathrm{I}}^i$ kanbans can be found between RIQ_i and the output buffer area, we should have $b_i - k_i + n_{\mathrm{I}}^i$ virtual kanbans at VSS_i. Now, consider case (ii). Since $k_i - n_{\mathrm{I}}^i > b_i$, all virtual kanbans are in use between RIQ_i and the output area. Hence, $n_{\mathrm{VB}}^i = 0$. ■

The above Lemma provides the basis for a *reverse mapping* from the arrival rates at the virtual bulletin board to that at the real bulletin board in an analysis of the nested CQNs. Since n_{VB}^i is *uniquely* mapped onto n_{I}^i when $n_{\mathrm{I}}^i < k_i - b_i$, we have

$$n_{\mathrm{VB}}^i = n_{\mathrm{I}}^i - k_i + b_{\mathrm{I}} \tag{4}$$

$$\lambda_{\mathrm{I}}^i(n_{\mathrm{I}}^i) = \lambda_{\mathrm{VB}}^i(n_{\mathrm{VB}}^i)\forall n_{\mathrm{I}}^i < k_i - b_{\mathrm{I}}. \tag{5}$$

However, $n_{\mathrm{VB}}^i = 0\ \forall n_{\mathrm{I}}^i \geqslant k_i - b_i$. Nevertheless, both the real and virtual kanbans always travel together within the virtual CQN. Consequently, the arrival rate at the virtual bulletin board when $n_{\mathrm{VB}}^i = 0$ is the same as the arrival rates at the real bulletin board for any $n_{\mathrm{I}}^i \geqslant k_i - b_i$. Applying this, we determine

$$\lambda_{\mathrm{I}}^i(n_{\mathrm{I}}^i) = \lambda_{\mathrm{VB}}^i(0)\ \ \forall n_{\mathrm{I}}^i \geqslant k_i - b_i. \tag{6}$$

The mapping from $v_{\mathrm{I}}^i(n_{\mathrm{I}}^i)$ to $\lambda_{\mathrm{VQ}}^i(n_{\mathrm{VQ}}^i)$ is termed the *forward mapping* of rates (equations 1–3), while the mapping from $\lambda_{\mathrm{VB}}^i(n_{\mathrm{VB}}^i)$ to $\lambda_{\mathrm{I}}^i(n_{\mathrm{I}}^i)$ is termed the *reverse mapping* of rates (equations 4–6). The forward and reverse mappings are used in an alternating manner within an iterative framework for the analysis of subsystem $1 < i < N$. Finally, putting all the above pieces together, we present the iterative algorithm for any intermediate subsystem under general blocking as follows.

Algorithm: General Blocking Subsystem Analysis (GBSA)

Step 0: {Initialization}
Consider subsystem i. Fix parameters $\lambda_u^i(n_u^i)$, $n_u^i = 0, \ldots, b_{i-1}$ and $\lambda_d^i(n_d^i)$, $n_d^i = 0, \ldots, k_{i+1}$. Initialize $\lambda_{\mathrm{I}}^i(n_{\mathrm{I}}^i)$, $n_{\mathrm{I}}^i = 0, \ldots, k_i$ to some initial values.

Step 1: {Solve $\mathrm{SS}_{(i-1)i}$}
Solve the Markov model of $\mathrm{SS}_{(i-1)i}$ (Di Mascolo et al. 1996). Determine the throughput rates $v_{\mathrm{I}}^i(n_{\mathrm{I}}^i)$, $n_{\mathrm{I}}^i = 0, \ldots, k_i$.

Step 2: {Determine arrival rates at virtual queue: Forward Mapping}
Determine $\lambda_{\mathrm{VQ}}^i(n_{\mathrm{VQ}}^i)$, $n_{\mathrm{VQ}}^i = 0, \ldots, k_i - b_i$ from $v_{\mathrm{I}}^i(n_{\mathrm{I}}^i)$, $n_{\mathrm{I}}^i = 0, \ldots, k_i$ from the forward mapping equations (1)–(3).

Step 3: {Solve the virtual CQN}
Solve the virtual CQN using the iterative approximation scheme given in Di Mascolo et al. (1996), by fixing $\lambda^i_{\mathrm{VQ}}(n^i_{\mathrm{VQ}})$, $n^i_{\mathrm{VQ}} = 0, \ldots, k_i - b_i$ at values determined from step 2 and $\lambda^i_d(n^i_d)$, $n^i_d = 0, \ldots, k_{i+1}$ at values already fixed in Step 0. Determine $\lambda^i_{\mathrm{VB}}(n^i_{\mathrm{VB}})$, $n^i_{\mathrm{VB}} = 0, \ldots, b_i$ from this analysis.

Step 4: {Determine arrival rates of real kanbans at $\mathrm{SS}_{(i-1)i}$: Reverse Mapping}
Determine $\lambda^i_{\mathrm{I}}(n^i_{\mathrm{I}})$, $n^i_{\mathrm{I}} = 0, \ldots, k_i$ from $\lambda^i_{\mathrm{VB}}(n^i_{\mathrm{VB}})$, $n^i_{\mathrm{VB}} = 0, \ldots, b_i$ from the reverse mapping equations (4)–(6).

Step 5: {Convergence Test}
Determine whether the recomputed λ^i_{I} values are within ϵ of their old values. If they are within the chosen interval, stop; else return to Step 1.

Algorithm GBSA solves the problem of analyzing a subsystem by partitioning it into two components: $\mathrm{SS}_{(i-1)i}$ and the virtual CQN. Note that these two components basically constitute the real CQN underlying the subsystem. Algorithm GBSA completes the analysis of the nested CQN system by embedding the virtual CQN analysis within the real CQN analysis, using the mappings between $\mathrm{SS}_{(i-1)i}$ and the virtual CQN. Finally, the overall analysis of the entire BOQN system under general blocking is completed using the framework presented earlier, with algorithm GBSA for the analysis of all subsystems $1 < i < N$, and the method of Di Mascolo et al. (1996) for subsystems 1 and N.

1.3 Manufacturing and Minimal Blocking Systems

Manufacturing blocking systems are easily analyzed by using the algorithm for general blocking developed above, by simply setting $b_i = 1$ for any cell i where manufacturing blocking is involved. The minimal blocking systems have been addressed in Di Mascolo et al. (1996).

1.4 Communication Blocking Systems

In this protocol, service on an input part in a cell can begin if and only if there is a space in the input queue at the following cell. In the kanban version, this translates to $b_i = 0$ at cell i where communication blocking is used. Consequently, service on an awaiting part can begin if and only if: (i) a space is available in the input area, and (ii) a kanban is available at the bulletin board of the following cell. Clearly, setting $b_i = 0$ will destroy the nested CQN structure of general blocking systems developed earlier. Therefore, a special nested CQN structure is developed for communication blocking.

The decomposition structure of communication blocking systems is shown in Figure 4. To begin with, consider any subsystem $1 < i < N - 1$. This subsystem is also modeled along the lines of intermediate subsystems in the general blocking case, using the concept of nested CQNs. We introduce the virtual synchronization station and the virtual kanbans as before. In addition, we also introduce another set of kanbans termed *Communication Blocking (CB)-kanbans* with a *Communication Blocking Bulletin Board (CBB)* at subsystem i. The bulletin board is appended to $\mathrm{SS}_{(i-1)i}$, and a part can leave $\mathrm{SS}_{(i-1)i}$ if and only if it is matched with a real kanban and a CB-kanban. The triplet joins the virtual queue at VSS_i, and is matched with a virtual kanban to enter RIQ_i. At this stage, the input part carries three kanbans with it. After service completion, the CB-kanban is detached and returned immediately to CBB_i. The completed part proceeds to the output area of $\mathrm{SS}_{i(i+1)}$ with the other two kanbans. The part flow from $\mathrm{SS}_{i(i+1)}$ to the next cell and the return of the real and virtual kanbans to their respective bulletin boards are the same as in the general blocking model. In this model, subsystem i is assigned $(k_i - 1)$ CB-kanbans, and b_i is set equal to 1. While the above decomposition structure is uniformly used for all subsystems $i = 2, \ldots, N - 2$, subsystem 1, $(N - 1)$ and N are modeled as shown in Figure 4.

The CB-kanbans are a kind of virtual kanbans themselves. They are specifically intended to model the flow control process in communication blocking. In fact, communication blocking is closely related to manufacturing blocking, and it is illustrative to derive the modeling logic from a comparison of the two protocol models shown in Figure 5. We develop this analysis as follows.

Consider a cell i with k_i waiting spaces. The communication blocking protocol specifies that: (i) cell i is blocked if there are exactly k_{i+1} parts in the next cell $i + 1$, and (ii) any part on which service is completed at cell i should immediately be transferred to cell $i + 1$. In comparison, the manufacturing blocking protocol specifies that cell i is blocked if: (i) there are exactly k_{i+1} parts in the next cell $i + 1$, and (ii) there is one part at cell i, on which service is completed and retained in its current buffer. Based on the above, it can easily be seen that: (i) the input waiting area at cell i under communication blocking consists of the output buffer at $\mathrm{SS}_{(i-1)i}$, the virtual queue at cell i, and RIQ_i, and (ii) the input waiting area under manufacturing blocking consists of only the virtual queue at i and RIQ_i in the subsystem configurations. This is shown in Figure 5. While the general blocking model ensures that there can be at most k_i parts in the input area of cell i under manufacturing blocking, the CB-kanban CQN and the output buffer of $\mathrm{SS}_{(i-1)i}$ together ensure this under communication blocking. In the communication blocking model, cell $i - 1$ is blocked when (i) a part exists in the output buffer of $\mathrm{SS}_{(i-1)i}$ and (ii) the $(k_i - 1)$ CB-kanbans exist between the virtual queue at i and RIQ_i. Finally, it is also easy to observe that both the CB-kanban and virtual kanban CQNs are nonblocked. Based on the above analysis and Theorem 1, the following theorem is obtained.

Theorem 2. *The nested CQN structure involving the real kanbans, virtual kanbans, and CB-kanbans yields the same time epochs and flow control as in communication blocking. Furthermore, each CQN is nonblocked.*

We now develop an analysis of the communication blocking subsystems using a set of mappings as before. Let $\lambda^i_{\mathrm{CB}}(n^i_{\mathrm{CB}})$, $n^i_{\mathrm{CB}} = 0, \ldots, k_i - 1$ denote the state-dependent arrival rates at CBB_i. Further, we now have two downstream arrival processes at $\mathrm{SS}_{i(i+1)}$. Let $\lambda^i_{\tilde{d}}(n^i_{\tilde{d}})$, $n^i_{\tilde{d}} = 0, \ldots, (k_{i+1} - 1)$ denote the state-dependent downstream arrival rates at CBB_{i+1} of $\mathrm{SS}_{i(i+1)}$. The other arrival and throughput rate

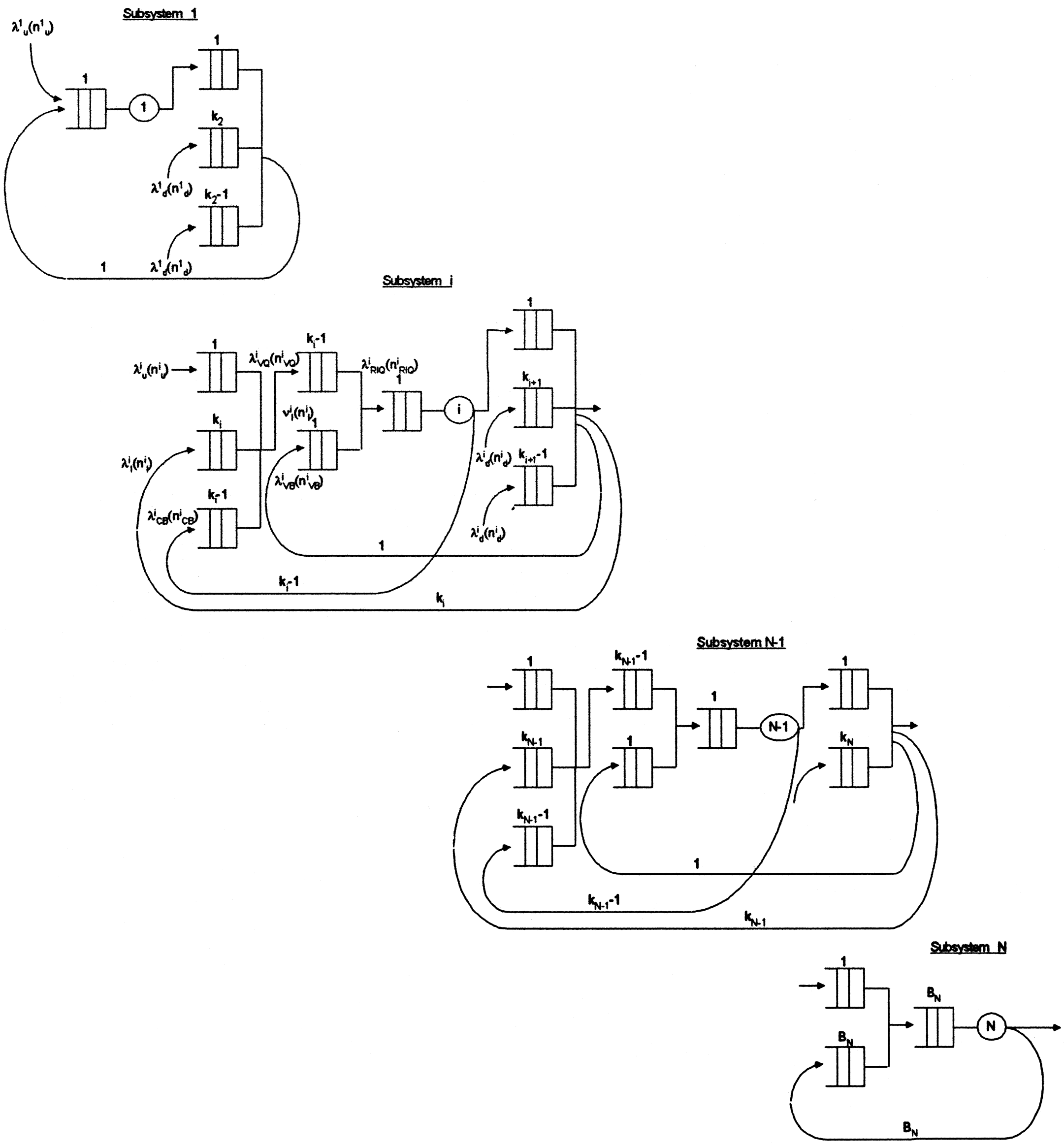


Figure 4. Communication blocking decomposition.

parameters are the same as before, and these parameters are indicated in Figure 4. The subsystem analysis begins with $SS_{(i-1)i}$, which is modeled as a Markov chain fed by three arrival processes as follows. Let (n_u^i, n_I^i, n^i_{CB}) define a state of the $SS_{(i-1)i}$ Markov chain. Note that n_u^i is equal to either 0 or 1. When $n_u^i = 0$, $n^i_{CB} = n_I^i - 1$ when $n_I^i = k_i$, and $(n^i_{CB} = n_I^i)$ or $(n^i_{CB} = n_I^i - 1)$ when $n_I^i < k_i$. Similarly, when $n_u^i = 1$, the only feasible configurations are $(n_I^i = n^i_{CB} = 0)$ and $(n_I^i = 1, n^i_{CB} = 0)$. With these conditions the state-transition diagram of the Markov chain is shown in Figure 6. This Markov chain yields a closed form solution as follows. Let P_{ijk} denote the steady-state probability of state (i, j, k). These probabilities are obtained using the following recursive path:

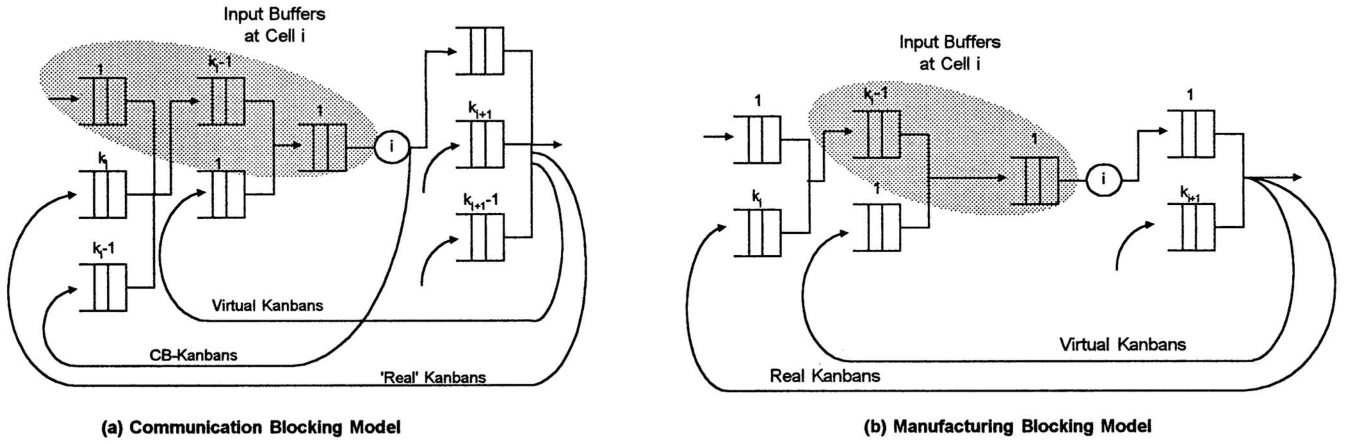
$$P_{0,k_i,k_i-1} = C \tag{7}$$

$$P_{0,k_i-2,k_i-2} = \frac{P_{0,k_i-1,k_i-2}\{\lambda_u^i(0) + \lambda^i_{CB}(0,k_i-1,k_i-2)\}}{\lambda_I^i(0,\ k_i-2,\ k_i-2)} \tag{8}$$

$$P_{0,k_i-1,k_i-1} = \frac{\lambda_u^i(0)C}{\lambda_I^i(0,\ k_i-1,\ k_i-1)} \tag{9}$$

$$P_{0,k_i-1,k_i-2} = \frac{P_{0,k_i-1,k_i-1} \cdot \lambda_u^i(0)}{\lambda^i_{CB}(0,\ k_i-1,\ k_i-2)} \tag{10}$$

$$P_{100} = \frac{P_{110}\lambda^i_{CB}(110)}{\lambda_I^i(100)} \tag{11}$$



(a) Communication Blocking Model

(b) Manufacturing Blocking Model

Figure 5. Modeling logic in communication and manufacturing blocking.

The equilibrium probabilities determined as above are used to map the throughput rates of $SS_{(i-1)i}$ to the input arrival rates at the virtual queue as follows. First, note that Lemma 1 is still valid in this case. Now consider that case when $n^i_u = 0$, $1 < n^i_{\mathrm{I}} \leq k_i$ and $1 \leq n^i_{\mathrm{CB}} < k_{i-1}$. The values of n^i_{VQ} in this case are in the discrete interval between 0 and $k_i - 3$. Further, in this case, there is always a kanban and a CB-kanban available when a part arrives at the output buffer in $SS_{(i-1)i}$. Consequently, the arriving part would immediately acquire a kanban from each set and move on to the virtual queue. Hence, we have from the above,

$$\lambda^i_{\mathrm{VQ}}(n^i_{\mathrm{VQ}}) = \lambda^i_u(0),\ 0 \leq n^i_{\mathrm{VQ}} \leq k_i - 3. \qquad (12)$$

Next, we consider the case $n^i_{\mathrm{I}} = n^i_{\mathrm{CB}} = 0$, $n^i_u = 0$ or 1. In this case, $n^i_{\mathrm{VQ}} = k_i - 1$. Clearly, there can be no arrival into the virtual queue unless a part moves out of it. Hence,

$$\lambda^i_{\mathrm{VQ}}(k_i - 1) = 0. \qquad (13)$$

Finally, consider the case when $n^i_{\mathrm{I}} = 1$. This case represents three possible states in the Markov model of $SS_{(i-1)i}$: (0, 1, 0), (1, 1, 0), and (0, 1, 1). Further, $n^i_{\mathrm{VQ}} = k_i - 2$ in this case. Using the equilibrium probabilities, we estimate the arrival rate here as follows:

$$\lambda^i_{VQ}(k_i - 2) = \frac{P_{110}\lambda^i_{CB}(110) + P_{011}\lambda^i_u(0)}{P_{110} + P_{011} + P_{010}} \qquad (14)$$

Note that the state (0, 1, 0) does not contribute to the arrival rate in the above equation since it requires a concurrent arrival of a part and a CB-kanban for an output to occur.

Now consider the reverse mapping case as in general blocking. Again, Lemma 2 holds, and equations (4)–(6) hold in the reverse determination of $\lambda^i_{\mathrm{I}}(n^i_{\mathrm{I}})$, $0 \leq n^i_{\mathrm{I}} \leq k_i$. The arrival of CB-kanbans to CBB_i follows the service pattern at server i. Consequently, if $v^i_{\mathrm{RIQ}}(n^i_{\mathrm{RIQ}})$, $n^i_{\mathrm{RIQ}} = 0, 1$ are the throughput rates at server i,

$$\lambda^i_{\mathrm{CB}}(n^i_{\mathrm{CB}}) = v_i(1),\ 0 < n^i_{\mathrm{CB}} < k_i - 1 \qquad (15)$$

$$\lambda^i_{\mathrm{CB}}(k_i - 1) = 0 \qquad (16)$$

Putting all these together, the iterative algorithm for the analysis of an intermediate subsystem under communication blocking is as follows.

Algorithm: Communication Blocking Subsystem Analysis (CBSA)

```
Step 0:  {Initialization}
         Consider subsystem $i$. Fix parameters $\lambda^i_u(n^i_u)$,
         $n^i_u = 0, 1$, $\lambda^i_d(n^i_d)$, $n^i_d = 0, \ldots, k_{i+1}$ and $\lambda^i_{\bar{d}}(n^i_{\bar{d}})$,
         $n^i_{\bar{d}} = 0, \ldots, k_{i+1} - 1$. Initialize $\lambda^i_{\mathrm{I}}(n^i_{\mathrm{I}})$, $n^i_{\mathrm{I}} =$
         $0, \ldots, k_i$ and $\lambda^i_{\mathrm{CB}}(n^i_{\mathrm{CB}})$, $n^i_{\mathrm{CB}} = 0, \ldots, k_i - 1$ to
         some initial values.
Step 1:  {Solve $SS_{(i-1)i}$}
         Solve the Markov model of $SS_{(i-1)i}$.
Step 2:  {Determine arrival rates at virtual queue:
         forward mapping}
         Determine $\lambda^i_{\mathrm{VQ}}(n^i_{\mathrm{VQ}})$, $n^i_{\mathrm{VQ}} = 0, \ldots, k_i - 1$ using
         the forward mapping equations (12)–(14).
Step 3:  {Solve the virtual CQN}
         Solve the virtual CQN problem as in the
         general blocking case. Use the Markov model
         for the communication blocking
         synchronization stations in the analysis of
         $SS_{i(i+1)}$. Determine $\lambda^i_{\mathrm{VQ}}(n^i_{\mathrm{VQ}})$, $n^i_{\mathrm{VQ}} = 0, 1$, and
         $v_i(1)$.
Step 4:  {Determine arrival rates at $SS_{(i-1)i}$: reverse
         mapping}
         Determine $\lambda^i_{\mathrm{I}}(n^i_{\mathrm{I}})$, $n^i_{\mathrm{I}} = 0, \ldots, k_i$ from equations
         (4)–(6). Determine $\lambda^i_{\mathrm{CB}}(n^i_{\mathrm{CB}})$, $n^i_{\mathrm{CB}} = 0, \ldots, k_i - 1$
         from equations (15)–(16).
Step 5:  {Convergence test}
         Determine whether the recomputed $\lambda^i_{\mathrm{I}}$ and $\lambda^i_{\mathrm{CB}}$
         values are within the chosen interval. If they
         are, then stop; else, return to Step 1.
```

While the above algorithm is applied in the analysis of subsystems $1 < i < N - 1$, the remaining subsystems are analyzed as follows. Subsystem $N - 1$ is also analyzed using algorithm CBSA, except that the Markov model for synchro-

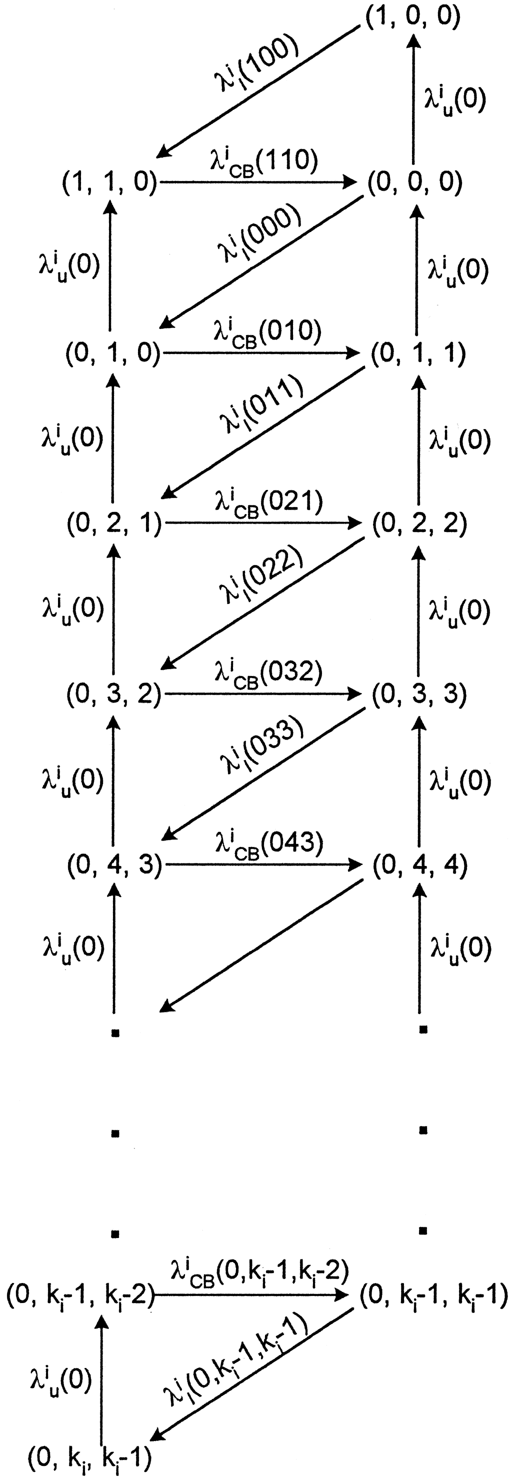


Figure 6. Markov chain of $SS_{(i-1)i}$ under communication blocking.

nization stations given in Di Mascolo et al. (1996) is used in the analysis of $SS_{(N-1)N}$. This is because there are no CB-kanbans in subsystem N (see Figure 4). The subsystem 1 and N are analyzed as in the previous models, since they require no special designs for communication blocking.

2. Blocked Closed Queueing Networks

The BCQN systems are analyzed using an *encapsulation and decomposition* framework as presented in Figure 7. Consider a BCQN system composed of N cells in tandem as shown in Figure 7(a). Let K denote the number of pallets in circulation. To begin, we introduce a synchronization station SS_{01} before cell 1. The station SS_{01} consists of a *pallet buffer* area to hold the returning pallets from the system, and a *bulletin board* for the kanbans in cell 1. Since the supply of raw material is saturated, the returning kanbans at cell 1 can be assumed to pick up the necessary raw material and wait at the bulletin board. When a kanban at the bulletin board is matched with a pallet, they are immediately transferred to the input queue at call 1. As before, we let $a_1 = b_1 = k_1$ and $a_N = k_N$, and any type of blocking can be used in the line. The service cells between the endpoints of this configuration are encapsulated and treated as a BOQN system as shown in Figure 7(b). Now, let $n_I^1 = 0, \ldots, k_1$, $n_p = 0, \ldots, K$ and $n^N = 0, \ldots, k_N$ denote the number of kanbans at SS_{01}, number of pallets at SS_{01}, and number of input parts awaiting service at cell N, respectively. Correspondingly, let $\lambda_I^1(n_I^1)$, $\lambda_p(n_p)$, and $\lambda_N(n^N)$ denote their state-dependent arrival rates. Let $v_N(n^N)$ denote the throughput rate of cell N. We develop an analysis of the encapsulated and decomposed BCQN system as follows.

First, note that the system comprising of SS_{01}, the encapsulated BOQN within, and cell N is a nonblocking CQN with the circulating pallets defining the loop. This follows from the nonblocking transfers out of SS_{01}, the nonblocking decompositions of the encapsulated system under any type of blocking protocol, and the nonblocked transfers in and out of cell N. Consequently, the resulting system configuration can be analyzed as a Markov chain with the following state space: $\{\langle n_p, n_I^1, n^N\rangle \mid n_p = 0, \ldots, K, n_I^1 = 0, \ldots, k_1, n^N = 0, \ldots, k_N\}$. Further, note that not all possible combinations of the three variables yield feasible states. The constraints on the possible combinations are as follows.

$$0 \leqslant n_p \leqslant K - k_1 \tag{17}$$

$$n_p n^1{}_I = 0 \tag{18}$$

$$\{K - n_p\} - \{k_1 - n_I^1\} - n_N \leqslant \sum_{i=2}^{N-1} k_i \tag{19}$$

Constraint (17) is straightforward, constraint (18) specifies that when $n_p > 0$, $n_I^1 = 0$, and when $n_I^1 > 0$, $n_p = 0$. This follows from the nonblocked transfer of a pallet and a kanban out of SS_{01} as soon as they are matched. Constraint (19) stipulates that the maximum number of pallets that could be in circulation among cells 2 through $N - 1$ at any time cannot exceed their cumulative maximum holding capacity. Figure 8 presents the maximum possible state space under these constraints for the underlying Markov chain. Figure 9 illustrates the state space for a specific example where $N = 3$, $k_1 = k_2 = k_3 = 3$, and $K = 6$.

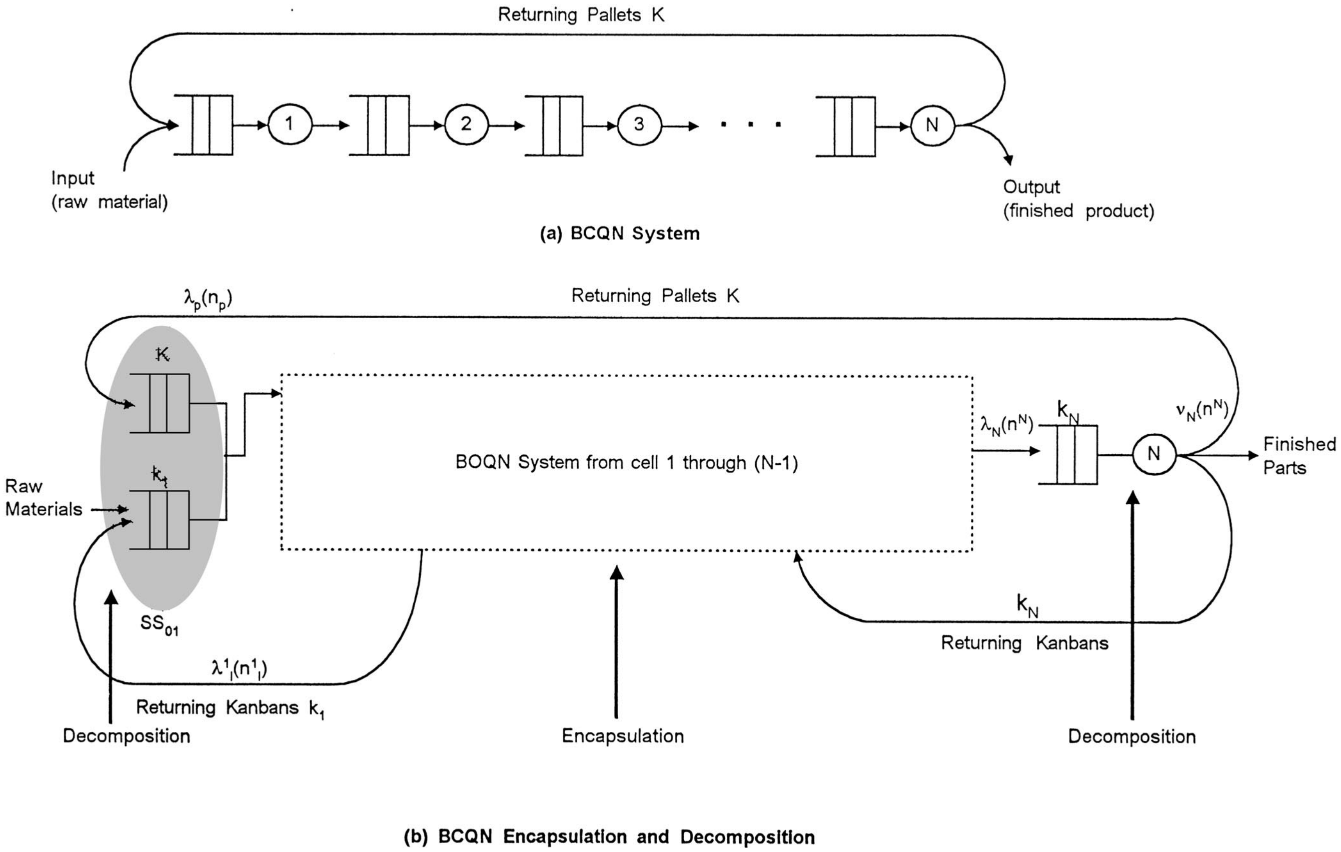


Figure 7. Encapsulation and decomposition framework for BCQN systems.

The above Markov chain does not yield a convenient closed form solution as in the BOQN systems, due to its structure. However, the transition-probability matrix is fairly sparse, and its solution in practical instances is quite inexpensive. This is discussed in the following section. After obtaining the equilibrium probabilities from the Markov solution, the arrival rates λ^1_I, λ_N, and the throughput rate v_N, the arrival rates $\lambda_p(n_p)$ are computed as follows.

$$\lambda_p(n_p) = \frac{\sum_{n^N} P(n_p, n^1_I, n^N) \cdot v_N(n^N)}{\sum_{n^N} P(n_p, n^1_I, n^N)}, \ \forall n_p > 0 \quad (20)$$

$$\lambda_p(0) = \frac{\sum_{n^1_I} \sum_{n^N} P(0, n^1_I, n^N) v_N(n^N)}{\sum_{n^1_I} \sum_{n^N} P(0, n^1_I, n^N)} \quad (21)$$

where P(n_p, n^1_I, n^N) is the equilibrium probability of sate $\langle n_p, n^1_I, n^N \rangle$. Linking all these together, we develop an approximation framework for BCQN systems as follows.

Step 0: {Initialization}
Initialize $\lambda_p(n_p)$, $n_p = 0, \ldots, K - k_1$, to some initial values.

Step 1: {Solve BOQN}
Incorporate SS$_{01}$ within subsystem 1. Hence, $\lambda_p(n_p)$ values provide the upstream rate parameters. Solve the BOQN problem for the line from cell 1 to cell N. Depending on the type of blocking involved, use the appropriate algorithm from the BOQN framework. Determine $\lambda^1_I(n^1_I)$, $n^1_I = 0, \ldots, k_1$, and $\lambda_N(n^N)$, $v_N(n^N)$, $n^N = 0, \ldots, k_N$ from this analysis.

Step 2: {Solve Markov Model}
Solve the Markov model $\langle n_p, n^1_I, n^N \rangle$ using the $\lambda^1_I(n^1_I)$, $\lambda_N(n^N)$ parameters determined from above. Determine $\lambda_p(n_p)$, $n_p = 0, \ldots, K - k_1$ from equations (20), (21).

Step 3: {Convergence test}
Determine whether the recomputed $\lambda_p(n_p)$ values are within a ϵ of their previous values. If so, stop; else, return to Step 1.

3. Computational Results

Detailed computational investigations on the proposed algorithms under the unified framework have been carried out. The focus of these studies has been on the accuracy of the approximations and the computational effort involved. The algorithms have been evaluated using benchmark com-

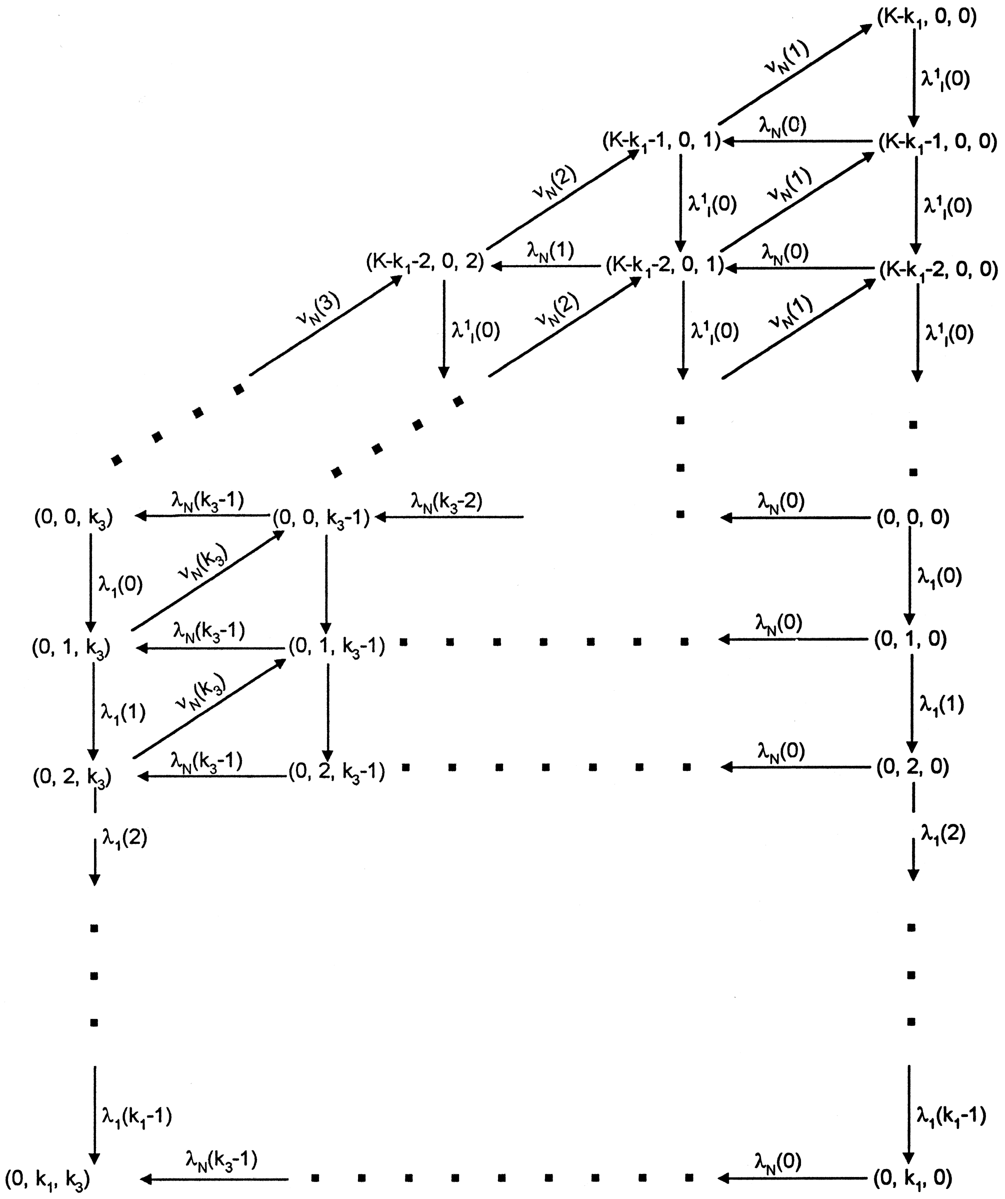


Figure 8. Maximum state space Markov model.

parisons with simulation results. The algorithms have been programmed in Fortran and implemented on a SUN/UNIX server with a 248 MHz SUNW, UltraSPARC-II CPU. The simulation models have been developed in the SIMAN simulation language (Pegden et al. 1995) and implemented on a 300 MHz IBM/PC. The algorithms on the UNIX server are fully portable to the PC environment. We present our results by organizing them into BOQN and BCQN studies as follows.

3.1 BOQN Studies

The experimental studies on BOQN systems are based on a fully crossed statistical design involving the following parameters and their associated levels:

(1) Blocking Protocol, BP: {Communication, Manufacturing, General}
(2) Number of cells, N: {5, 6, 7, 8, 9, 10}
(3) Number of kanbans in each cell, k: {5, 8, 10, 12}
(4) Number of output buffers in each cell $1 < i < N$, under general blocking, b_i: $\{2, \ldots, k-1 \mid k\}$
(5) Coefficient of variation in service time distribution, $CV^2 = \{0.5, 1, 2\}$

In total, the above design yielded 72 trials in communication and manufacturing blocking each, and 504 trials in general blocking. Simulations of the line under all these configurations have been carried out for the production of 50,000 parts after a warmup of 5,000 initial parts. Further, the simulations

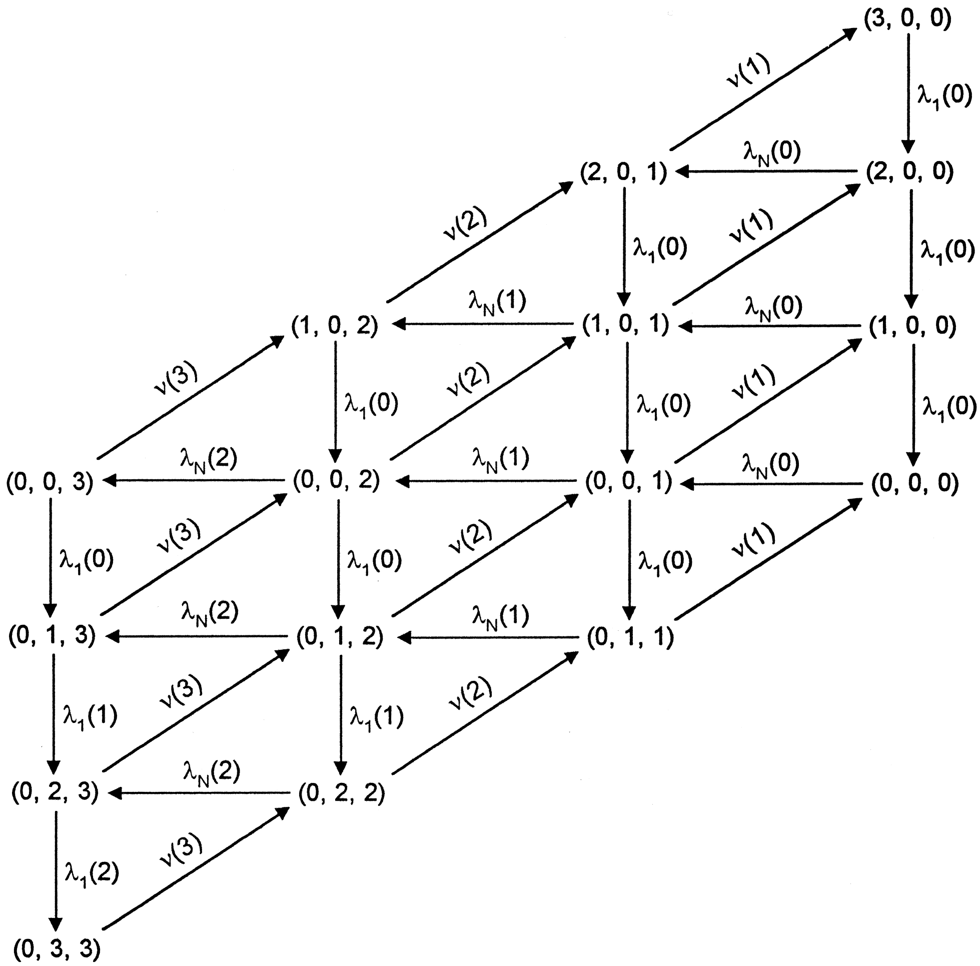


Figure 9. An illustration of the Markov state space where $N = 3$, $K = 6$, and $k_1 = k_2 = k_3 = 3$.

yielded steady performance characteristics under this run length and warmup conditions. The chosen warmup is 10% of the simulation run length. We also experimented with larger warmup conditions and it did not make any significant change to these characteristics. The simulations required between 10 minutes to more than an hour of CPU time on the PC, depending on the line parameters. Compared to this, the average CPU time required by the approximation algorithms on the UNIX server has been 0.01 CP seconds, with a maximum of 0.04 CP seconds. All algorithms converged in the trials conducted with $\epsilon = 0.001$. The value of ϵ is the upper limit of $|(\text{old value} - \text{new value})/\text{old value}|$ of arrival and departure rates at each stage at convergence. The chosen value $\epsilon = 0.001$ is therefore extremely small and signifies stable convergence conditions. Any further reduction in ϵ may not alter the convergence values significantly. Further, increasing ϵ may also not help as it may affect the quality of the estimates without yielding any computational advantages, since the computational load of the algorithms with $\epsilon = 0.001$ is already minimal. The agreements between the approximation and simulation results have been measured in terms of two performance measures:

$$\%WIP = \frac{\begin{array}{c}Total\ WIP(Approximation)\\ -\ Total\ WIP(Simulation)\end{array}}{Total\ WIP(Simulation)} \times 100$$

$$\%THP = \frac{\begin{array}{c}Throughput(Approximation)\\ -\ Throughput(Simulation)\end{array}}{Throughput(Simulation)} \times 100$$

where WIP stands for work-in-process inventory. The results of these studies are summarized for representative configurations in Table I. Similar results have been obtained in the rest of the trials as well.

Table I also includes the confidence intervals for %WIP and %THP given as percentages of the simulation average WIP and throughput, respectively. The confidence intervals are used to determine the statistical precision of the simulation results. The confidence interval represents the range of values in which the true average will be included with a probability of 95%. For example, for $N = 5$, $k = 5$, and $b =$

Table I. Open Queueing Network Results for $CV^2 = 1$

N	*k*	*b*	% WIP	Confidence Interval (%)	% THP	Confidence Interval (%)
COMMUNICATION			−3.33	1.11	−4.36	2.81
5	5	1	−2.20	0.35	−0.61	1.75
5	5	2	−1.07	1.17	0.87	2.77
5	5	3	−0.79	1.04	−3.15	2.55
MINIMAL			−2.71	0.63	−1.87	2.93
COMMUNICATION			−0.88	1.19	−1.32	2.27
5	10	2	−2.20	0.31	−0.35	2.01
5	10	5	−4.00	0.65	−1.67	1.65
5	10	8	−4.93	1.18	−1.72	2.33
MINIMAL			1.09	0.49	0.54	2.41
COMMUNICATION			−9.10	0.52	−7.21	2.34
7	5	1	−3.21	0.86	−1.04	1.97
7	5	2	−1.36	0.41	0.82	3.18
7	5	3	0.43	1.03	−3.47	1.55
MINIMAL			−4.59	0.44	−2.16	1.91
COMMUNICATION			−4.95	0.60	−1.80	2.91
7	10	2	−0.42	1.05	−0.27	3.07
7	10	5	−7.65	1.08	0.38	1.97
7	10	8	−0.55	1.10	−0.91	1.84
MINIMAL			3.31	1.19	0.60	2.47
COMMUNICATION			−17.84	0.42	−11.25	2.34
10	5	1	−1.61	0.36	0.92	1.81
10	5	2	2.36	0.51	0.77	1.82
10	5	3	−3.17	1.07	−3.42	2.64
MINIMAL			−6.31	0.68	−2.30	2.18
COMMUNICATION			−11.51	0.34	−2.87	3.42
10	10	2	−0.56	1.07	−0.70	2.57
10	10	5	−5.29	1.23	−0.65	2.10
10	10	8	−2.05	1.04	−1.22	2.50
MINIMAL			0.34	0.65	−0.75	2.19

1, we are 95% sure that the true average WIP is within 0.35% of the average WIP found through simulation.

These results indicate that the throughput approximation on the average is within 1.5%, 2.3%, and 5.6% of that of simulation for general, manufacturing, and communication blocking systems, respectively. Similarly, the WIP approximation is on the average within 2.2%, 2.5%, and 8.1% of that of simulation for these protocols. We extensively tested the approximation schemes for serial systems that commonly occur in practice. In general, the approximation algorithms performed very well. The results shown in Table I are representative of all values of *N*, *k*, and CV^2 tested. The deviation ranges are comparable when these parameters are varied. In all our experiments, the deviations that we observed from simulation results in general blocking systems are between 0.24% and 3.92% for throughput and between 0.43% and 7.8% for WIP. Therefore, from a qualitative point of view, the approximations are close to actual simulation results within acceptable limits (roughly, <4% deviation in throughput and <8% deviation in WIP) for general blocking systems under the range of configuration parameters tested. The deviations we observed from simulation results in communication blocking systems are between 1.54% and 4.34% for throughput and between 0.67% and 18.23% for WIP. Similarly, the deviations of the manufacturing blocking approximation are between 0.64% and 4.22% for the throughput and between 1.03% and 8.97% for the WIP. The only situation where the deviations from the simulation results are sizable is when the number of output buffers is very low, as in communication blocking. These results are consistent with those observed by Di Mascolo et al. (1996) with minimal blocking systems. The deviations are basically due to the assumption of state-dependent Markov arrival processes in the decomposition framework, which may not be accurate when the number of kanbans as well as the output buffers are very small. We also experimented with unbalanced serial systems by randomly generating the mean service times in the configurations tested using the same convergence criteria. We obtained similar performance results with the algorithms in these cases as well. From these analyses, we conclude that: (i) the approximations work well for general blocking systems of any configuration for serial systems that

Table II. Minimal Blocking Closed Queueing Network Results

N	k	Mu	CV^2	K	% THP	Confidence Interval (%)	% WIP	Confidence Interval (%)
5	5	5	1	12	−4.34	4.51	−2.29	7.60
5	5	5	1	15	−2.57	3.24	0.45	8.72
5	5	5	1	18	−1.91	4.96	−0.30	7.10
5	5	5	1	20	−3.06	2.45	−3.03	7.63
5	5	5	1	22	−0.96	3.98	−0.09	6.63
5	5	5	1	24	−1.16	2.44	−2.06	5.76
5	5	5	1	25	−1.58	4.29	−2.25	8.49
10	10	5	1	50	−2.11	2.96	−0.53	5.81
10	10	5	1	60	−1.76	3.07	−3.21	8.96
10	10	5	1	70	−1.35	2.60	−2.07	7.42
10	10	5	1	80	−0.72	2.89	−0.76	8.60
10	10	5	1	90	−0.62	4.75	−3.48	5.11

Table III. Communication Blocking Closed Queueing Network Results

N	k	Mu	CV^2	K	% THP	Confidence Interval (%)	% WIP	Confidence Interval (%)
5	5	5	1	6	−11.32	2.45	−4.69	6.68
5	5	5	1	8	−10.37	3.55	−6.47	7.13
5	5	5	1	10	−13.40	2.45	−5.28	6.41
5	5	5	1	12	−9.59	2.68	−3.81	6.81
5	5	5	1	14	−3.27	3.02	3.13	5.70
10	10	5	1	40	−3.35	3.24	−4.13	5.67
10	10	5	1	45	−2.35	2.55	−3.42	8.33
10	10	5	1	50	−1.82	4.82	−2.49	5.15
10	10	5	1	55	−1.81	4.47	−5.81	5.56
10	10	5	1	60	−1.51	4.75	−6.14	7.76

commonly occur in practice, and (ii) the approximations for communication blocking systems are not as good as those of general blocking, but nevertheless give fairly good assessments, given the computational effort involved. In this analysis, we have chosen to compare the performance of the proposed algorithms with results of simulations incorporating actual system parameters. Hence the deviations in the approximations from the simulation results are the real deviations from actual system performance. Given the above performance results, we also conclude that the approximation framework yields reliable and accurate results at almost negligible computational effort for practical systems of reasonable size. Although it will be illustrative to compare the proposed approach with other approximations in the literature, we are not aware of any approach for approximating general blocking systems. Most of the literature on queueing system approximation is concerned with minimal and manufacturing blocking systems (see Govil and Fu 1999 for a survey of the state of the art in queueing systems in manufacturing). These blocking protocols really do not yield a comparable basis, as they are special cases of the general blocking systems considered in this research.

3.2 BCQN Studies

The same experimental design as in the BOQN systems is used in this case as well, except with the addition of the following parameter: number of pallets (K). This parameter is varied at four levels in each instance of the other parametrical combinations: $\{3k, 3(k+1), 3(k+2), 3(k+3)\}$. Note that $N \cdot k$ is an upper bound on K. The above levels yield different flow rates within a line as determined by the number of pallets.

The results of the BCQN studies are summarized for representative configurations in Tables II, III, and IV. Similar results have been obtained with the rest of the trials as well. The BCQN results are comparable to those of BOQN systems. The throughput approximations are on the average within 3.3%, 3.8%, and 5.9%, and the WIP approximations within 2.8%, 3.3%, and 3.9% of those simulations for general, manufacturing, and communication blocking systems, respectively. The average CPU time for a BCQN approximation is about 1 CPU second on the UNIX server. BCQN systems require a greater computational load than BOQN systems, and this is mostly due to the matrix inversion required in solving the BCQN Markov model. Nevertheless,

Table IV. General Blocking Closed Queueing Network Results

N	k	b	Mu	CV^2	K	% THP	Confidence Interval (%)	% WIP	Confidence Interval (%)
5	5	1	5	1	15	−4.75	4.15	4.11	7.65
5	5	1	5	1	18	−2.57	3.16	−1.64	8.91
5	5	1	5	1	21	−2.38	4.13	−2.10	7.22
5	5	1	5	1	24	−1.92	4.75	−1.35	6.39
5	5	2	5	1	15	−6.43	4.26	0.65	7.84
5	5	2	5	1	18	−2.76	3.48	−1.11	5.79
5	5	2	5	1	21	−2.04	4.50	−1.16	6.10
5	5	2	5	1	24	−1.58	4.99	−2.76	6.99
5	5	3	5	1	15	−6.74	4.15	1.98	6.29
5	5	3	5	1	18	−3.96	4.59	1.41	5.90
5	5	3	5	1	21	−1.76	2.27	0.56	8.30
5	5	3	5	1	24	−0.97	4.61	0.04	5.85
10	10	2	5	1	50	−0.94	2.30	3.82	7.51
10	10	2	5	1	60	0.16	2.85	1.32	6.22
10	10	2	5	1	70	0.87	3.50	−1.37	6.55
10	10	2	5	1	80	−0.70	2.71	−0.45	6.40
10	10	5	5	1	50	−0.70	3.26	−2.49	8.38
10	10	5	5	1	60	−0.68	3.94	−1.60	5.42
10	10	5	5	1	70	−0.56	3.92	−2.04	8.43
10	10	5	5	1	80	−1.04	3.68	−2.69	7.03
10	10	8	5	1	50	−3.29	3.24	1.63	8.10
10	10	8	5	1	60	−2.33	2.54	2.93	5.37
10	10	8	5	1	70	−1.23	4.93	−2.33	7.31
10	10	8	5	1	80	−0.48	4.68	−3.09	8.06

the total CPU time requirement is very small, compared to the simulation alternative. Given the quality of the approximations as demonstrated from the above results and the minimal CPU loads, the proposed framework is viable, accurate, and efficient in approximating BOQN as well as BCQN systems under any general blocking protocol.

4. Conclusion

In this paper, we developed a unified framework for approximating open as well as closed queueing serial systems under a variety of blocking protocols. Such network systems predominate in several application areas such as cellular manufacturing, telecommunications, and distributed control systems. The intrinsic complexities in the exact analysis of such systems have focused considerable attention in the literature on both approximation methods and simulation techniques. While much of the approximation literature is concentrated on open queueing networks, research on closed queueing systems is relatively restricted.

Di Mascolo et al. (1996) present an approximation algorithm for open tandem queues under the minimal-blocking kanban configuration. We extend and generalize their work to open queueing systems under any general blocking protocol, and synthesize the framework to approximate closed queueing systems under any blocking protocol as well. Using a unified framework for approximation, we develop specific decomposition strategies and approximation algorithms for a variety of system configurations and control structures. Our approach is founded on two principles: equivalence between tandem queues and kanban systems under any general blocking protocol, and the decomposition of a line into isolated subsystems with an iterative analysis linking the subsystems. All the approximation algorithms developed in this work are derived from these principles, and can be cast from the unified framework for general system approximation.

The proposed algorithms have been extensively tested. Our results indicate that the proposed framework yields robust, reliable, and accurate estimates of system characteristics such as throughput and WIP in a wide range of system configurations. Furthermore, the computational effort involved is minimal. Together, these two performance measures indicate the attractiveness of the proposed framework in the analysis of complex queueing systems, especially compared with the popular and widely used simulation alternative.

The proposed unified framework presents a set of highly useful tools of analysis to queueing-systems designers, whether they are kanban-based or not. A major concern in the design of queueing control systems is the tradeoff between throughput and WIP, and a designer may be required to evaluate several alternate design scenarios in terms of these tradeoffs before arriving at a good design. While simulation is mostly used for this purpose, it is often expensive. It therefore restricts the number of alternatives a designer may consider. Furthermore, recent developments in the general blocking area suggest that general blocking systems could provide superior performance characteristics in terms of throughput and WIP over the traditional manufacturing and minimal blocking systems (Ramesh et al. 1997a, b). While general blocking is promising, it adds another significant dimension to the design space by enabling the selective control of either the input or output buffers at a cell. Consequently, a designer is faced with an exponentially increasing array of design options, indicating the need for quick and accurate assessments of queueing system configurations. In this respect, the proposed framework addresses this need.

We see several possible avenues of future research. First, the analyses of tandem BOQN and BCQN systems developed here can be generalized to any network queueing structure. In this case, the models developed above can be generalized using visit ratios and other routing algorithms that may be used in the various service nodes of a network. This generalization particularly applies to the X.25 and ISDN (Integrated Services Digital Networks) packet switching wide area networks in telecommunications (Stallings 1997). Second, the above analyses can be extended to multiple product flows in group-technology-based cellular manufacturing systems and priority queueing systems in a variety of service areas. Finally, the equivalence of tandem queues and kanban systems can be exploited to analyze a variety of control options such as common buffer areas and transfer lines, and arrival behaviors such as balking, reneging, group arrivals, etc. In some cases, the Markov analyses may be generalized as renewal processes. While these research extensions are theoretically attractive, they are also practically relevant and important. We are currently investigating some of these ideas.

Acknowledgements

We wish to thank Maria DiMoscolo for several useful discussions during the initial stages of this work. We are also indebted to Sid Jagabandhu for his help with modeling some of the blocking systems. Finally, we thank the two anonymous reviewers for their the critical comments, which have greatly contributed to this paper.

References

Akyildiz, I.F. 1987. Exact product form solutions for queueing networks with blocking. *IEEE Trans. Comput.* **1** 121–126.

Baskett, F., K.M. Chandy, R.R. Muntz, F.G. Palacios. 1975. Open, closed, and mixed networks of queues with different classes of customers. *J. ACM* **22** 249–260.

Baynat, B., Y. Dallery. 1993. A unified view of product-form approximation techniques for general closed queueing networks. *Perform. Eval.* **18** 205–224.

Bruell, S.C., G. Balbo. 1980. *Computational Algorithms for Closed Queueing Networks.* Elsevier North-Holland, Amsterdam.

Buzacott, J.A. 1988. Kanban and MRP Controlled Production System. Working Paper, Department of Management Science, University of Waterloo, Ontario, Canada.

Buzen, J.P. 1973. Computational algorithms for closed queueing networks with exponential servers. *Communications of the ACM* **16** 527–531.

Cheng, D.W., D.D. Yao. 1994. Tandem queues with general blocking: a unified model and comparison results. *Discrete Event Dynamic Systems* **2** 207–234.

Dallery, Y. 1990. Approximate analysis of general open queueing networks with restricted capacity. *Perform. Eval.* **11** 209–222.

Dallery, Y., Y. Frein. 1993. On decomposition methods for tandem queueing networks with blocking. *Opns. Res.* **41** 386–399.

Di Mascolo, M., Y. Frein, Y. Dallery. 1996. An analytic method for performance evaluation of kanban controlled production systems. *Opns. Res.* **44** 50–64.

Gordon, W.J., G.F. Newell. 1967a. Cyclic queueing systems with restricted queues. *Oper. Res.* **15** 266–278.

Gordon, W.J., G.F. Newell. 1967b. Closed queueing systems with exponential servers. *Oper. Res.* **15** 254–265.

Govil, M.K., M.C. Fu. 1999. Queueing theory in manufacturing: a survey. *J. of Manf. Sys.* **18** 214–240.

Marie, R. 1980. Calculating equilibrium probabilities for $\lambda(n)/C_k/1/N$ queues. *Perform. Eval. Rev.* **9** 117–125.

Mitra, D., I. Mitrani. 1990. Analysis of a kanban discipline for cell coordination in production lines, I. *Mgmt. Sci.* **35** 1548–1566.

Mitra, D., I. Mitrani. 1991. Analysis of a kanban discipline for cell coordination in production lines, II: stochastic demands. *Opns. Res.* **36** 807–823.

Onvural, R.O. 1990. Survey of closed queueing networks with blocking. *ACM Computing Surveys.* **22** 83–121.

Pegden, C.D., R.E. Shannon, R.P. Sadowski. 1995. *Introduction to Simulation Using SIMAN,* 2nd ed. McGraw-Hill, New York.

Perros, H.G. 1994. *Queueing Networks with Blocking: Exact and Approximate Solutions.* Oxford University Press, New York.

Perros, H.G., A. Nilsson, Y.G. Liu. 1988. Approximate analysis of product form type queueing networks with blocking and deadlock. *Perform. Eval.* **8** 19–39.

Ramesh, R., S.Y. Prasad, M. Thirumurthy. 1997a. Flow control in kanban multicell manufacturing, I. A structured analysis of general blocking design space. *Int. Journal of Prod. Research* **35** 2327–2342.

Ramesh, R., S.Y. Prasad, M. Thirumurthy. 1997b. Flow control in kanban multicell manufacturing, II. Design of control systems and experimental results. *Int. Journal of Prod. Research* **35** 2413–2427.

Stallings, W. 1997. *Data and Computer Communications,* 5th ed. Prentice-Hall, New Jersey.

Van Dijk, N.M., H.C. Tijms. 1986. Boxma, Cohen, and Tijms, eds. *Insensitivity to Two Node Blocking Models With Applications, Teletraffic Analysis and Computer Performance Evaluation.* Elsevier North-Holland, Amsterdam. 329–340.

Zipkin, P. 1989. A kanban-like production control system: analysis of simple models. Research Working Paper No. 89-1, Graduate School of Business, Columbia University, New York.