Throughput Analysis in Closed Queueing Networks with Finite Buffers

by
R.O. Onvural
and
H.G. Perros

Center for Communications and Signal Processing
Department of Electrical Computer Engineering
North Carolina State University

March 1987

CCSP-TR-87/2

ABSTRACT

The behavior of throughput in closed queueing networks with finite buffers is studied as the number of customers changes from one to the capacity of the network. Based on the results obtained, an approximation algorithm is developed to calculate the throughput of large closed queueing networks under two different types of blocking mechanisms. Validation tests show that the algorithm is fairly accurate.

Key words:closed queueing networks, blocking, deadlock

1.INTRODUCTION

In recent years, there has been a growing interest in the development of computational methods for the analysis of queueing networks with finite buffers. This is primarily due to a growing need to model actual systems which have finite capacity resources. An important feature of systems with finite buffers is that a server may become blocked when the capacity limitation of the destination queue is reached. Various blocking mechanisms have been considered in the literature so far. These blocking mechanisms arose out of various studies of real life systems. A discussion on these different blocking mechanisms can be found in Onvural and Perros [7].

In this paper, we will study such closed queueing networks under two types of blocking mechanisms, namely type 1 and type 2. In Type 1 blocking mechanism, a customer declares its destination after it completes its service. If upon service completion, a customer at node i attempts to enter node j and finds it full, then the customer will be forced to wait in front of server i until a space becomes available at node j. Server i remains blocked for this period of time, and it can not serve any other customer waiting in its queue. If more than one server is blocked by the same node, then these servers will get unblocked in a first-blocked-first-unblocked fashion.

Due to the blocking mechanism described above, and due to the fact that these N nodes are arbitrarily interconnected, it is possible that deadlocks may occur. For instance, assume that node i is blocked by node j. Now it is possible that a customer in node j may, upon completion of its service, choose to go to node i. If node i is at that time full, then deadlock will occur. In this paper, it is assumed that deadlocks are detected immediately and resolved by instantaneously exchanging blocking units. This may violate the first-blocked-first-unblocked rule described above. For instance, let us assume that nodes i and k are blocked by node j in that order. That is, if a departure occurs from node j the blocked customer from node i will enter node j first. Now, let us assume that the departing unit from node j chooses node k as its destination, and, that node k is full at that moment. This causes a deadlock to occur, which is resolved by simultaneously exchanging the blocking units from nodes j and k. In view of this, the blocked customer from node k enters node j first while node i still remains blocked. Thus, the first- blocked-first-unblocked priority rule has been violated.

In Type 2 blocking mechanism, a customer in queue i declares its destination queue j just before it starts its service. If queue j is full the ith server becomes blocked. When a departure occurs from destination queue j, the ith sever becomes unblocked and the customer begins receiving service. While the server is blocked, the position in front of the server is occupied by the customer. In this type of blocking mechanism, the number of customers in the network should be such that deadlocks do not occur. This is due to the fact that, deadlocks can not be resolved in this type of blocking mechanism unless some customers are removed from the system.

A closed queueing network with finite buffers under Type 1 blocking is shown to have product form solution when the network has exactly two nodes (see, Akyildiz [1], Diehl [3], Perros [9]). Onvural and Perros [8] showed that a closed queueing network under type 1 blocking, has a product form solution when the number of customers,K, is equal to the capacity of the smallest buffer plus one. Gordon and Newell [4] considered Type 2 blocking mechanism and introduced the concept of holes in closed tandem networks. They proved that if the number of holes in the network is less than or equal to the minimum buffer capacity, then the network has a product form solution. Persone and Grillo [10] compared the average visiting time and utilization of a node in symmetrical closed queueing networks under three different blocking mechanisms.

Approximation algorithms for analyzing closed queueing networks with finite buffers have been proposed by Suri and Diehl [11], and Akyildiz [1,2]. In particular, Suri and Diehl studied closed tandem queueing networks with finite capacities in which the first queue has an infinite capacity. Their algorithm is restricted to only tandem configuration and validation tests were restricted to networks with small populations. The algorithm works for both blocking mechanisms defined in this paper. Akyildiz [1] finds an equivalent non-blocking network, which has the same or almost equal number of states as the blocking network. Both systems have approximately the same behavior, and the throughput of both systems is approximately the same . In Akyildiz [2], another approximation for blocking queueing networks is introduced using mean value analysis. The approximation is based on the modification of mean residence times due to the blocking events that occur in the network. He considered only Type 1 blocking mechanism.

Throughput is defined next. The behavior of the throughput as the number of customers varies is discussed in Sections 2.1, 2.2 and 2.3. Based on these results, approximation algorithms are developed and validated in Section 3.

2.THROUGHPUT OF CLOSED QUEUEING NETWORKS

In this paper, we will consider closed queueing networks consisting of N nodes and K customers. Each node consists of a single queue served by a server with an exponentially distributed service time with rate μ_i , i=1...,N. B_i is the capacity of node i including the service space in front of the server. A customer upon completion of its service at node i attempts to enter destination node j with probability p_{ij}, i=1,..,N; j=1,..,N.

Throughput of a node is defined as the rate at which customers depart from that node. Gordon and Newell [5] showed that closed queueing networks with infinite buffer capacities (CQN-I) have a product form solution. Using this property, several well known algorithms have been developed in the literature to calculate the throughput of a network (i.e., MVA, Convolution Algorithm, etc).

Let $\beta_i(K)$ and $\beta(K)$ be the throughput of a node and throughput of CQN-I with K customers in it. Then, by definition $\beta_i(K)=\{1-P_i^K(0)\}\mu_i$ where $P_i^K(0)$ is the probability that node i is empty and μ_i is the service rate at node i. Furthermore $\beta_i(K)=\beta(K)e_i$ where e_i is the mean number of visits a customer makes to node i and is given by:

$$e_i = \sum_{j=1}^{N} p_{ji} e_j, i=1,\dots,N; \quad \textit{with} \quad e_j=1 \quad \text{for} \quad \textit{some} \quad j. \tag{1}$$

One efficient algorithm to calculate the throughput of a CQN-I is given in Akyildiz [1]

$$\beta(K)=\frac{K}{\sum_{m=1}^{K}\sum_{j=1}^{m-1}\beta(K-j)\sum_{i=1}^{N}x_i^m} \tag{2}$$

where $x_i=e_i/\mu_i$ is the relative utilization of node i. When restrictions are imposed on buffer capacities, product form solutions are, in general, not available.

For closed queueing networks with finite buffers (CQN-B), let $\lambda_i(K)$ and $\lambda(K)$ be the throughput of node i and the throughput of the network respectively. Then we have: $\lambda_i(K)=\{1-P_i^K(0)-P_i^K(b)\}\mu_i$ where $P_i^K(0)$ and $P_i^K(b)$ are the probabilities that node i is empty and blocked respectively given there are

K customers in the network. Also, $\lambda_i(K)=\lambda(K)e_i$, $i=1,\ldots,N$ where e_i is given by (1).

Clearly, $\lambda_i(K)$ depends on the parameters of the network. In figures 2 and 3, we give two examples of $\lambda_3(K)$ as K changes from 1 to $M(=\sum_{i=1}^{3} B_i)$ for the network shown in figure 1.

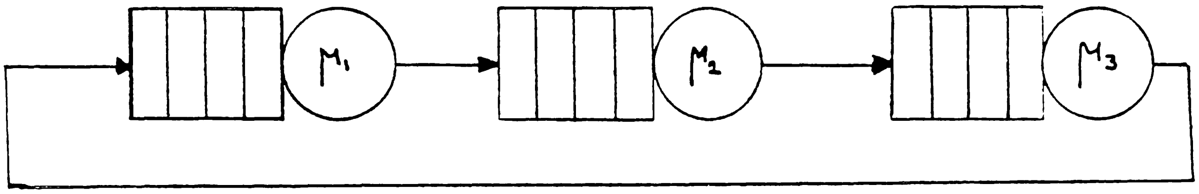

Figure 1: A three-node cyclic network

We note that in figures 2 and 3, $\lambda_3(K)$ increases as K increases until it reaches a maximum value, λ^*, for some K^*, $K^* \epsilon \{L:\lambda_3(L) \geq \lambda_3(i), i=1,\ldots,M\}$ where the set can be a singleton or can have more than one element. For $K>K^*$, $\lambda_3(K)$ is non-increasing on K. The behavior of these graphs can be explained as follows: $P_3^K(0)$ decreases, as K increases, until it reaches 0 at $K=M-B_3+1$. This value of K is such that the number of holes (i.e. free spaces) in the queueing network, M-K, is equal to B_3-1. That is, in all states queue 3 will contain at least one customer. For $M-B_3+1\leq K \leq M$, $P_3^K(0)=0$. $P_3^K(b)=0$ for $1\leq K\leq B_1$ and non-decreasing beyond B_1. Hence, $P_3^K(0)+P_3^K(b)$

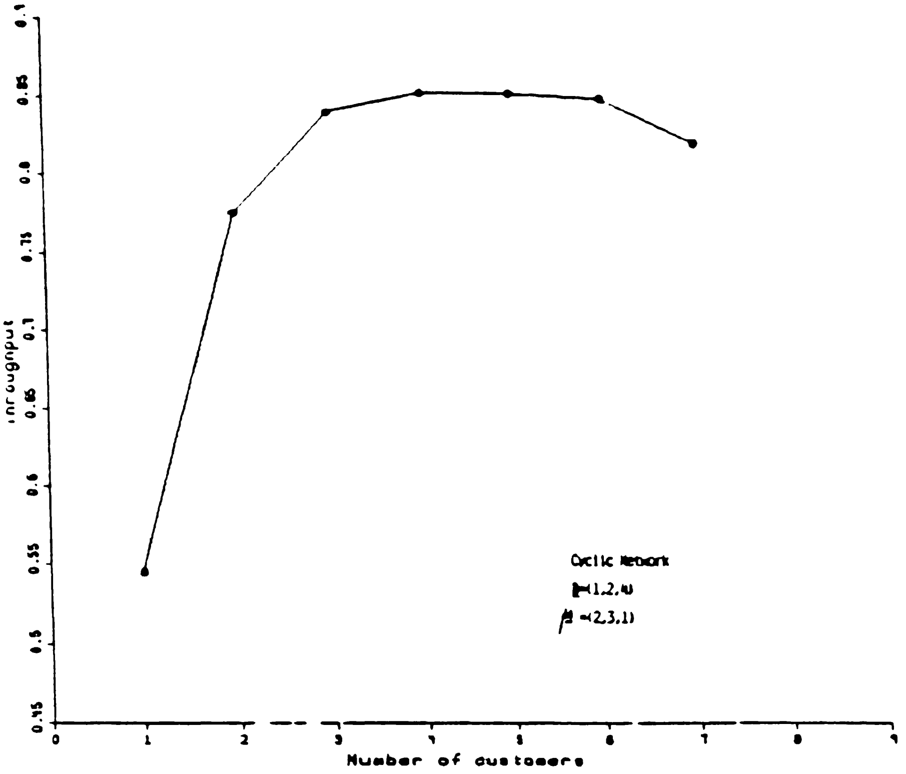

Fig. 2:Throughput vs # of customers-

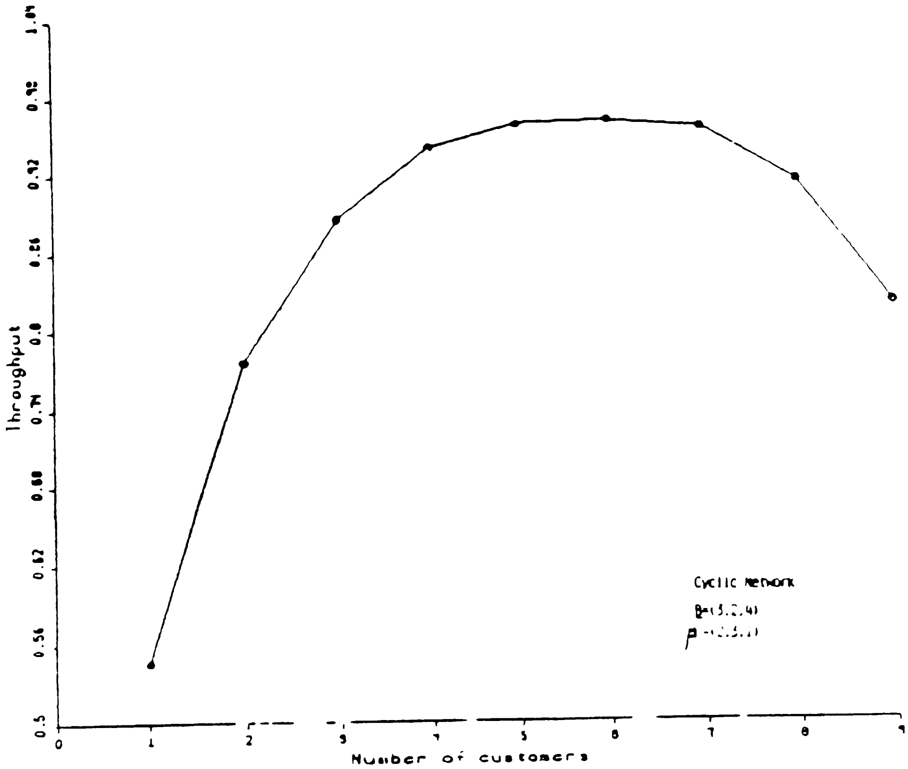

Fig.3:Throughput vs # of customers

is non-increasing in the interval $[1,B_1]$ and non-decreasing in $[M-B_3+1,M]$. It is not clear what happens in the interval $[B_1+1,M-B_3]$. Empirically, we have observed that there is a point $K^* \epsilon [B_1+1,M-B_3]$ where $P_3^K(0)+P_3^K(b)$ continues to be non-increasing in $[B_1+1,K^*]$ and non-decreasing in $[K^*+1,M-B_3]$.

2.1. CQN-B under Type 1 blocking

Theorems 1 through 3 and corresponding corollaries presented below can be found in Onvural and Perros [8] hence they are not proved here. These results present the equivalencies between CQN-Bs with respect to the buffer capacities and the number of customers in the network. They also help us to understand the behavior of throughput as a function of the number of customers in the network.

THEOREM 1: Let $\lambda^* = \max_K\{\lambda(K)\}$, $n = \min_{i=1,..,N}\{B_i\}$ and $M = \sum_{i=1}^{N} B_i$. Then

$$\beta(n+1) \le \lambda^* \le \beta(M - \max_{i=1,..,N}\{B_i\}+1)$$

COROLLARY 1:Let K^* be such that $\lambda^* = \lambda(K^*)$. Then,

$$\max_{i=1,..,N}\{\min_{j=1,..,N}\{B_j \text{ such that } p_{ij} \neq 0\}\} \le K^* \le M - \max_{i=1,..,N}\{B_i\}+1$$

THEOREM 2:Let $M = \sum_{i=1}^{N} B_i$, $B^* = \max_{i=1,...,N}\{B_i\}$

and $S=\{L: M-B^* +1 \le L \le B^* +1\}$. If $2B^* \ge M$ (i.e. S is not empty) then the network with K customers in it has the same steady state queue length distribution for all $K \in S$.

COROLLARY 2:If there exists a j such that $2B_j \ge M$ then increasing the buffer capacity of node j will not change the value of the maximum throughput.

THEOREM 3:Let μ_i, p_{ij} be the parameters of two closed queueing networks with buffer capacities B_i and C_i, i=1,..,N; j=1,..,N. If $\min_{i=1,\dots,N} B_i \ge l+1$ and $\min_{i=1,..,N} C_i \ge l+1$, $l \ge 0$, that is $0 \le l \le \min\{\min_{i=1,\dots,N} B_i-1, \min_{i=1,\dots,N} C_i-1\}$, then the two networks with $\sum_{i=1}^{N} B_i - l$ and $\sum_{i=1}^{N} C_i - l$ customers respectively have the same rate matrix.

COROLLARY 3:Let M be the total capacity of the network. Then, $\lambda(M)$ is independent of buffer capacities B_i, i=1,..,N.

Now, we will present another result for cyclic networks with $K=\sum_{i=1}^{N} B_i$ customers. Consider the cyclic network shown in Figure 1 and let $B_i=1$, i=1,2,3, and K=3. Furthermore, let X_i be the service time at node i and at time 0 all servers are busy working. Without loss of generality, assume $X_1 \le X_2 \le X_3$. At $t=X_3$, deadlock will occur. In this paper, we assume that deadlocks are detected immediately and resolved by exchanging blocking units. Hence at time X_3, the customer at node 1 will go to node 2, the customer at node 2 will go to node 3 and the customer at node 3 will go to node 1. At this point in time, all servers will start service and they will be busy working. The points at which all servers begin a new service, i.e. they all become busy working, are renewal points. In this example, the mean time between renewals is equal to $E[\max(X_1,X_2,X_3)]$.

THEOREM 4: In cyclic networks, if the number of customers in the network ,K, is equal to M $(=\sum_{i=1}^{N} B_i)$ then the throughput of the network is equal to $1/E[\max(X_1,X_2,\ldots,X_N)]$.

Proof: Let N(t) be the number of times all servers are busy working at or prior to time t. Then N(t) is a renewal process, and the mean time between renewals is equal to $E[\max(X_1,X_2,\ldots,X_N)]$ where X_i is distributed exponentially with rate μ_i, i=1,...,N. Consider any node i. Then a departure from this node will occur at renewal points. Hence, the mean time it takes a customer to depart from that node is equal to the mean time between renewals. Thus, the throughput of node i is equal to $1/E[\max(X_1,X_2,\ldots,X_N)]$. Furthermore, since X_is are distributed exponentially, we have

$$P(\max_{i=1,\ldots,N}(X_i)<t)=\prod_{i=1}^{N} P(X_i<t),$$

or $P(\max_{i=1,\dots,N}(X_i)<t)=\prod_{i=1}^{N}(1-e^{-\mu_i t})$

Hence, $E[max(X_1,\ldots,X_N)]=\int_0^{\infty}(1-\prod_{i=1}^{N}(1-e^{-\mu_i t}))\,dt$

2.2. CQN-B under Type 2 blocking

Gordon and Newell [4] introduced the concept of holes in cyclic networks under Type 2 blocking. Consider a cyclic network with N nodes. Let B_i, μ_i be the buffer capacity and the service rate of node i respectively, i=1,...,N, where the service times are distributed exponentially. Deadlocks do not occur in this blocking mechanism hence the number of customers in the network should be less than the capacity of the network, i.e. $K<\sum_{i=1}^{N}B_i$. Since the buffer capacity of node j is B_j, let us imagine that this node consists of B_j cells. If there are i_j customers at node j, then i_j of these cells are occupied and B_j-i_j are empty. We may say that these empty cells are occupied by holes. The number of holes in the system is equal to $\sum_{i=1}^{N}B_i$-K. As the customers move sequentially through the network, the holes execute a counter-sequential motion since each movement of a customer from the jth to the (j+1)st node corresponds to the movement of a hole in the opposite direction. Hence, they showed that two systems are dual.

LEMMA 1: Let B_i, μ_i be the parameters of a cyclic network under Type 2 blocking. Then the network with K customers has the same throughput as the network with $\sum_{i=1}^{N} B_i$-K customers. Hence, the maximum throughput is achieved at $K^* = \lfloor M/2 \rfloor$.

Theorem 5, presented below, is true for cyclic networks and is a generalization of Theorem 2 to Type 2 blocking mechanism. Furthermore, it can be generalized to arbitrary configurations provided that the maximum number of customers in the network is such that deadlocks can not occur.

THEOREM 5: Let $M = \sum_{i=1}^{N} B_i$, $B^* = \max_{i=1,..,N} \{B_i\}$ and $S=\{L: M-B^* \leq L \leq B^*+1\}$. If $2B^* \geq M$ (i.e. S is not empty) then the cyclic network with K customers in it has the same steady state queue length distribution for all $K \in S$.

Proof: This theorem was proved by Onvural and Perros [8] for Type 1 blocking. The same proof can also be used for Type 2 blocking for $M-B^* \leq K \leq \lfloor M/2 \rfloor$, and for $\lceil M/2 \rceil \leq K \leq M$ the proof follows from Lemma 1.

In Type 2 blocking mechanism, a server gets blocked if the destination queue of its customer, who is about to start service, is full. A variation of this blocking mechanism may be defined as follows:

Type 2^* blocking mechanism:A server will be blocked if at least one of its destination queues is full.

THEOREM 6: In CQN-B under Type 2^* blocking, if the network has a reversible routing matrix, then it has a product form joint queue length distribution.

Proof: If the network has a reversible routing matrix and infinite buffer capacities then it has a product form solution with state space S. Under Type 2^* blocking, the state space of the network, A, is a subset of S. Let q(j,k) be the rate of going from state j to state k in the network with infinite buffer capacities. Let

$$q'(j,k)=\begin{cases}0 & \text{if } j \in A \text{ and } k \in S-A \\ q(j,k) & \text{if } j \in A \text{ and } k \in A\end{cases}$$

then the process with state space A and rates $q'(j,k)$ is a truncated process and hence it has a product form solution (Kelly[7]). But, it can be verified that the truncated process is the queueing network under Type 2^* blocking. Therefore, CQN-B under Type 2^* blocking has a product form solution.

2.3.Equivalency of Type 2 and Type 1 Blocking Mechanisms

Let N1 be a cyclic network with parameters B_i and μ_i under type 1 blocking and N2 be a cyclic network with parameters C_i and μ_i under type 2 blocking, i=1,...,N; where N>2. In the following theorem, we prove that there exists an equivalency between Type 1 and Type 2 blocking mechanisms.

THEOREM 7: Let N1 and N2 be defined as above. Furthermore, let $C_i = B_i + 1$, i=1,...,N. Then, the two networks are equivalent for all K such that $1 \le K \le \min\{ \min_{i=1,\dots,N-1} (C_i + C_{i+1}), \min(C_N + C_1)\} - 1$.

Proof: Let $(b_1, \dots, b_N)$ be the state of N1 where b_i is the number of customers at node i. Furthermore, let $(b_1, \dots, b_i, b_{i+1}^*, \dots, b_N)$ denote that node i is blocked by node i+1. Likewise, let $(c_1, \dots, c_N)$ be the state of N2 where c_i is the number of customers at node i. Note that node i in N2 is blocked by node i+1 if $c_{i+1} = C_{i+1}$ and $c_i > 0$.

We observe that, for $1 \le K \le \min\{ \min_{i=1,\dots,N-1} (C_i + C_{i+1}), \min(C_N + C_1)\} - 1$, the two networks have the same number of states. Let us now apply the following transformations to the states of N1:

$(b_1, \dots, b_N) \Longrightarrow (b_1, \dots, b_N)$, if no node is blocked and

$(b_1, \dots, b_i, b_{i+1}^*, \dots, b_N) \Longrightarrow (b_1, \dots, b_i - 1, b_{i+1} + 1, \dots, b_N)$ if node i is blocked, i=1,...,N.

It can be easily shown that the set of the transformed states is in fact the state space of N2. Above transformation is one to one between the set of states associated with N1 and those of N2, and onto. Also, the transition rates to and out of a state in N1 and its associated state in N2 are the same. Hence N1 has the same rate matrix as N2.

3.APPROXIMATION ALGORITHMS

In this section, we will present approximation algorithms for analyzing cyclic queueing networks and the central server model under Type 1 and Type 2 blocking.

We, first, summarize the results of the last section as applied to the throughput of closed queueing networks with finite buffers. For $1 \leq K \leq L$, for some L, the network has a product form solution and the throughput is increasing in this interval as the number of customers increases. For $I \leq K \leq M$, for some I, $I \geq L$, throughput is decreasing as K increases. Let,

$$S_{K^*} = (K^* | \lambda(K^*) \geq \lambda(K), \ \ 1 \leq K \leq M)$$

Then, S_{K^*} is the set of number of customers for which the throughput is maximum. Furthermore, let us assume that the set S_{K^*} is a singleton, i.e. there is exactly one point, K^*, at which the throughput is maximum. One can easily find examples where this assumption does not hold (one immediate example may be a network with a node i where $2B_i \geq M$, i.e. Theorems 2 and 5). But, this assumption is made only for presentation purposes and it is not used in the approximation algorithms.

Estimating the throughput of a network as the number of customers varies from one to the capacity of the network can be seen as estimating the parameters of a curve that passes through the points $\lambda(1),\ldots,\lambda(L)$, $\lambda(K^*)$, and $\lambda(M)$. this curve can then be used to estimate the throughput at the unknown points , $L+1,\ldots,K^*-1$, and $K^*+1,\ldots,M-1$. Based on our empirical observations, we will assume that $\lambda(K)$ is non-decreasing in $[1,K^*]$ and non-increasing in $[K^*,M]$.

The main steps of the approximation procedures can be generalized as follows:

i) If there exists a node i with $2B_i \geq M$ then let $B_i = \lfloor M/2 \rfloor$

ii) Find K^*, and calculate $\lambda(K^*)$

iii) Calculate $\lambda(1),\ldots,\lambda(L)$, and $\lambda(M)$

iv) Estimate the parameters of the curve that passes through 1) $\lambda(1),\ldots,\lambda(L)$ and $\lambda(K^*)$ with a non-negative slope; 2)$\lambda(K^*)$ and $\lambda(M)$ with a positive slope.

v) Calculate the unknown throughput values from the equations of these curves.

At step i, we are using theorems 2 and 5 to eliminate some of the redundant points in the throughput curve. The maximum throughput is calculated using a numerical technique. The throughput for the points in step iii is calculated using one of the efficient algorithms for product form networks since there is no blocking.

Next, we will discuss, how these individual steps can be implemented for each type of blocking.

3.1.Cyclic networks under type 2 blocking

Consider a cyclic network, shown in Figure 1, under type 2 blocking. From Lemma 1, the throughput curve is symmetric, i.e.$\lambda(i)=\lambda(M-i)$; i=1,...,M-1, and maximum occurs at $\lfloor M/2 \rfloor$. Hence, we only need to calculate $\lambda(1),...,\lambda(\lfloor M/2 \rfloor$. But, $\lambda(1),...,\lambda(\min B_t)$ can be calculated efficiently, i.e. using Eq.(2), since the network has a product form solution. $\lambda(\lfloor M/2 \rfloor)$ can be calculated from the joint queue length distributions obtained by numerically solving the network with $\lfloor M/2 \rfloor$ customers. We first experimented with some interpolation algorithms, such as cubic splines and least squares method to estimate the throughput at unknown points using the throughput at known points. But these algorithms did not provide good approximations. This is due to the fact that we have few known points to use and the length of the interval between K^* and $\min B_t$ is usually large. Hence, the two end points do not provide much information for the points in between. The following equation was found to yield good approximations, as it takes into account the rate at which the throughput changes as the number of customers changes.

$$\lambda(i)=\lambda(i+1)-yx^{K^*-i} \tag{3}$$

where x and y can be determined as follows. For i=min B_t, we have:

$$\lambda(\min B_i)=\lambda(\min B_i+1)\text{-}yx^{K^*-minB_i}$$

$$=\lambda(\min B_i+2)\text{-}yx^{K^*-minB_i-1}\text{-}yx^{K^*-minB_i}$$

.

.

$$=\lambda^*\text{-}y\sum_{i=1}^{K-minB_i} x^{K^*-i}$$

Hence, $y=\dfrac{\lambda^*-\lambda(minB_i)}{\sum_{i=1}^{K-minB_i} x^{K^*-i}}$. Substituting y into Eq.(3), we have:

$$\lambda(\mathrm{i})=\lambda(\mathrm{i}+1)\text{-}x^{K^*-i}\,\frac{\lambda^*-\lambda(minB_i)}{\sum_{i=1}^{K-minB_i} x^{K^*-i}}$$

Now, let i=minB_i-1. Then;

$$\lambda(\min B_i\text{-}1)=\lambda(\min B_i)\text{-}x^{K^*-(minB_i-1)}\,\frac{\lambda^*-\lambda(minB_i)}{\sum_{i=1}^{K-minB_i} x^{K^*-i}}$$

$$=\lambda(minB_i)-\frac{x^{K^*-1-minB_i}(1-x)}{x-x^{K^*-minB_i+1}}(\lambda^*-\lambda(minB_i))$$

Since, the values of λ(minB_i), λ(minB_i-1), and λ^* are known, solving x can be treated as a fixed point problem. The unknown throughput points can now be calculated using Eq.(3) for i=K^*-1,K^*-2,...,minB_i+1. Note that, to calculate x, we assume that $\min B_i \geq 2$. If, $\min B_i = 1$ then, we have to solve the network with $\min B_i + 1$ customers numerically to calculate $\lambda(\min B_i + 1)$. The complete algorithm is given in Appendix A. We applied our algorithm to several cyclic networks with different buffer capacities and service rates. The number of nodes in the network was varied from 3 to 8. The results show that the algorithm is quite accurate. Of the 50 examples we run, the relative error percentage (i.e. 100*(exact throughput - app. throughput)/exact throughput) was less than 4%. A representative set of examples is given in Tables 1 to 10. The exact throughput was obtained numerically.

3.2.Cyclic networks under Type 1 blocking

Let B_i, μ_i be the parameters of a cyclic network and let $M = \sum_{i=1}^{N} B_i$. Then, to estimate the number of customers at which the throughput is maximum, we will use the equivalency of type 1 and type 2 blocking, presented in Theorem 7. In Theorem 7, we showed that a cyclic network under Type 1 blocking with buffer capacities B_i, i=1,...,N is equivalent to the same cyclic network under Type 2 blocking with buffer capacities B_i+1, i=1,...,N, for some range of number of customers in the network. But, in Type 2 blocking, the maximum throughput is achieved at $K^* = \left\lfloor \frac{\sum_{i=1}^{N} B_i + 1}{2} \right\rfloor$. Hence, in Type 1 blocking, we will assume that $K^* = \left\lceil \frac{M+N}{2} \right\rceil$-1 and solve the network numerically with K^* customers in order to calculate $\lambda(K^*)$. Furthermore, $\lambda(M)$ can be calculated using Theorem 4, and $\lambda(1),\ldots,\lambda(\min B_i+1)$ can be calculated from Eq.(2) since the network has a product form solution with K customers for $1 \le K \le \min B_i + 1$. The main steps of the algorithm are given in Appendix B.

The algorithm was applied to a variety of cyclic networks. The number of nodes was varied from 3 to 7. A representative set of examples is given in Tables 11 to 18. The exact throughput in each of the examples was obtained numerically. We note that the approximation results have low relative error.

3.3. The Central Server Model

Now consider the central server model shown below.

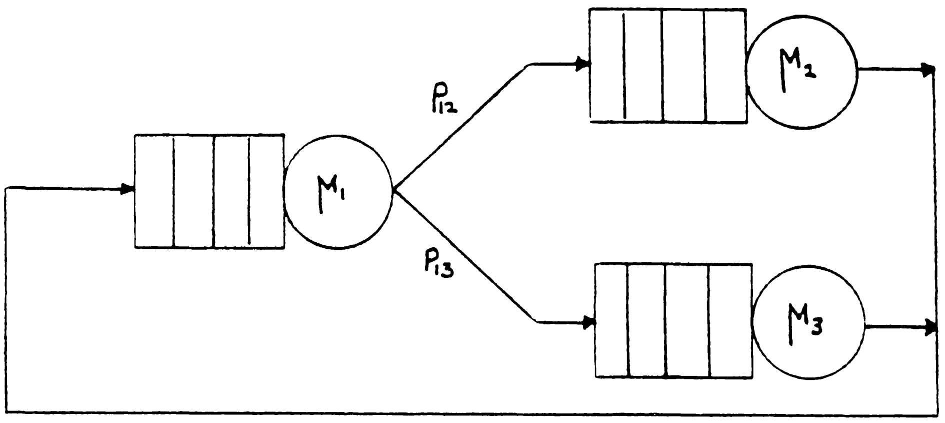

Figure 4:Central Server Model

The parameters of the network are B_i, μ_i, and p_{1i}, i=1,...,N with $p_{11}=0$.

Case 1: Let $B_1=\infty$ and $B_i<\infty$, i=2,...,N, and consider the network under Type 1 blocking. Furthermore, let $R=\sum_{i=2}^{N} B_i$. Since, $B_1=\infty$, we have that $2B_1>R$. Using Theorem 2, we conclude that the throughput of the network is constant for $K\geq R+1$. Noting that the maximum throughput is achieved at $K^*=R+1$, we applied the algorithm given in Appendix B to calculate $\lambda(\min B_i+2)$,..., $\lambda(R)$. A set of representative examples is given in Tables 19 to 25. The results on the exact throughput were obtained numerically. We note that the approximation results have a very low relative error.

Case 2: Let $B_1=\infty$ and $B_i<\infty$, i=2,...,N, and consider the network under Type 2 blocking. Furthermore, let $R=\sum_{i=2}^{N} B_i$. From Theorem 5, we conclude that the throughput of the network is constant for $K\geq R$. Hence, the maximum throughput is achieved at $K^*=R$. With this modification, we applied the algorithm presented in Appendix A to these networks with different service rates, routing probabilities and B_1s. The examples given in Tables 26 to 32 show that the algorithm is fairly accurate. The exact throughput given in these tables was obtained numerically.

Case 3: Let $B_1<\infty$, and $B_i=\infty$, i=2,...,N. Then the network has a product form solution under Type 2 blocking from Theorem 6.

Case 4: Let $B_1<\infty$, $B_i=\infty$, i=2,...,N and consider the network under Type 1 blocking. If, $p_{1i}=1/(N-1)$, i=2,...,N, then Onvural and Perros [8] showed that the network has a product form solution after the service rate at node 1 is modified. However, if $p_{1i}\neq 1/(N-1)$ then the network could not be shown to have a product form solution. Furthermore, our algorithm is not applicable, since, the number of customers at which the throughput is maximum is not readily available. We have also observed that, the throughput increases as the number of customers in the network increases.

Appendix A:Algorithm for Type 2 blocking

S1: Calculate $\lambda(i)$, i=1,..., $\min_{i=1,\dots,N} (B_i)$

S2: Solve the network numerically with $K^* = \left\lfloor \frac{M}{2} \right\rfloor$ customers and calculate $\lambda^* = \lambda(K^*)$

S3: Let DIF1:=$\lambda^* - \lambda(min(B_i))$
DIF2:=K^*-minB_i

Calculate x from the following equation:

$$\lambda(minB_i)) - \frac{DIF1 * x^{DIF2+1} * (1-x)}{x - x^{DIF2+1}} = \lambda(minB_i - 1)$$

S4: Let DIF=$\frac{DIF1*(1-x)}{x - x^{DIF2+1}}$, and A=$\lambda^*$

```
for i=($K^*$-1) downto (min$B_i$+1) do
  begin
    $\lambda$(i)=A-$x^{K^*-i}$*DIF
    A=$\lambda$(i)
    $\lambda$(i)=$\lambda$(i)+$c_i$*DIF
  end;
```

where

$$c_i = \begin{cases} K^* - i & i = 1, \dots, \lfloor Z \rfloor \\ c_{i+1} & \text{if } i = \lceil Z \rceil \text{ and } \lceil Z \rceil \neq \lfloor Z \rfloor \\ c_{i+1} - 1 & i = \lfloor Z \rfloor + 1, \dots, M-1 \end{cases}$$

and $Z = \dfrac{K^* - min(B_i)}{2}$

Appendix B:Approximation Algorithm for Type 1 Blocking

S1: Calculate $\lambda(M)$ using Theorem 4. Let $\lambda(M-1)=\lambda(M)+(\beta(3)-\beta(2))*0.975$

S2: Let $K^* = \left\lceil \dfrac{M+N}{2} \right\rceil$-1 and solve the network numerically with K^* customers to

calculate $\lambda^* = \lambda(K^*)$

S3: Let DIF1:=$\lambda^* - \lambda(min(B_i+1))$
DIF2:=K^*-minB_i-1

Calculate x from the following equation:

$$\lambda(minB_i+1)) - \frac{DIF1 * x^{DIF2-1} * (1-x)}{x^2 - x^{DIF2-1}} = \lambda(minB_i)$$

S4: Let DIF$=\dfrac{DIF1*(1-x)}{x^2 - x^{DIF2-1}}$,

Let A=λ^*

```
for i=($K^*$-1) downto (min$B_i$+2) do
  begin
    $\lambda$(i)=A-$x^{K^*-i}$*DIF
    A=$\lambda$(i)
    $\lambda$(i)=$\lambda$(i)+$c_i$*DIF
  end;
```

S5: Let DIF$=\dfrac{(\lambda^{*}-\lambda(M-1))*(1-x)}{x-x^{M-K^{*}}-1}$ and A$=\lambda^{*}$

```
for i=$K^{*}$ to M-2 do
 begin
   $\lambda$(i):=A-$x^{K^{*}}-i$*DIF
   A=$\lambda$(i)
   $\lambda$(i)=$\lambda$(i)+$d_i$*DIF
 end;
```

where

$$c_i=\begin{cases} K^{*}-i & i=1,\ldots,\lfloor Z\rfloor \\ c_{i+1} & \text{if } i=\lceil Z\rceil \text{ and } \lceil Z\rceil\neq\lfloor Z\rfloor \\ c_{i+1}-1 & i=\lfloor Z\rfloor+1,\ldots,M-1 \end{cases}$$

and Z$=\dfrac{K^{*}-min(B_i)-1}{2}$

$$\text{and } d_i=\begin{cases} K^{*}-i & i=1,\ldots,\lfloor Z\rfloor \\ d_{i-1} & \text{if } i=\lceil Z\rceil \text{ and } \lceil Z\rceil\neq\lfloor Z\rfloor \\ d_{i-1}-1 & i=\lfloor Z\rfloor+1,\ldots,M-1 \end{cases}$$

and Z$=\dfrac{M-K^{*}}{2}$

REFERENCES

[1] I.F. Akyildiz, "Analysis of Closed Queueing Networks with Blocking", CS Dept,85-022 (1985),Lousiana State University

[2] I.F. Akyildiz, "Mean Value Analysis for Blocking Queueing Networks", Manuscript

[3] G.W. Diehl, "A Buffer Equivalency Decomposition Approach to Finite Buffer Queueing Networks", Ph.D. Thesis, Eng. Sci., Harvard University, (1984)

[4] W.J. Gordon and G.F.Newell, "Cyclic Queueing Systems with Restricted Queue Lengths", Oper. Res.,15(1967), 266-278

[5] W.J. Gordon and G.F. Newell,"Closed Queueing Systems with Exponential Servers",Oper. Res.,15(1967),254-265

[6] F.P. Kelly, "Reversibility and Stochastic Networks", John Wiley and Sons, (1979)

[7] R.O. Onvural and H.G. Perros, "On Equivalencies of Blocking Mechanisms in Queueing Networks with Blocking", To appear in OR Letters

[8] R.O. Onvural and H.G. Perros, "Some Exact Results in Closed Queueing Networks with Finite Buffers", CCSP, NC State Univ, (1986).

[9] H.G. Perros,"A Survey of Queueing Networks with Blocking (Part I)", Dept. of CS,86-04 (1986),NC. State University

[10] V. De Nitto Persone and D. Grillo, "Managing Blocking in Finite Capacity Symmetrical Ring Networks", Third Int. Conf. on Data Comm. Sys. and Their Perf., (1987)

[11] R. Suri and G.W. Diehl, "A New 'Building Block' for Performance Evaluation of Queueing Networks with Finite Buffers", Proc. ACM Sigmetrics, (1984), 131-142

# cust	exact throughput	approximate throughput	relative error %
5	1.765	1.761	0.2
6	1.331	1.327	0.25
7	1.363	1.361	0.1
8	1.376	1.374	0.08

Table 1: B=(5,4,5,4) , μ=(3,4,5,2)

# cust	exact throughput	approximate throughput	relative error %
3	0.851	0.845	0.75
4	0.916	0.906	1.08
5	0.942	0.935	0.74
6	0.948	0.946	0.3

Table 2: B=(2,4,3,5), μ=(2,1,3,4)

# cust	exact throughput	approximate throughput	relative error %
3	0.854	0.869	1.8
4	0.956	1.011	2.6
5	1.078	1.113	3.3
6	1.138	1.153	1.3
7	1.171	1.173	0.1

Table 3: B=(2,4,3,3,4), μ=(2,2,2,2,2)

# cust	exact throughput	approximate throughput	relative error %
3	1.39	1.388	0.1
4	1.557	1.557	0
5	1.655	1.665	0.65
6	1.71	1.71	0
7	1.735	1.733	0.13

Table 4: B=(2,4,3,3,4), μ=(5,3,4,6,2)

# cust	exact throughput	approximate throughput	relative error %
3	1.261	1.25	0.9
4	1.443	1.422	1.5
5	1.556	1.535	1.4
6	1.624	1.61	0.8
7	1.662	1.656	0.44
8	1.685	1.694	0.28
9	1.696	1.694	0.16
10	1.702	1.7	0.08

Table 5: B=(4,4,3,3,6,2), μ=(5,3,4,6,2,4)

# cust	exact throughput	approximate throughput	relative error %
3	0.619	0.625	1.07
4	0.697	0.71	1.8
5	0.749	0.766	2.3
6	0.783	0.798	1.96
7	0.806	0.813	0.88
8	0.819	0.82	0.2

Table 6: B=(2,3,4,2,4,3), μ=(3,2,4,1,5,1)

# cust	exact throughput	approximate throughput	relative error %
3	0.764	0.757	0.92
4	0.858	0.846	1.4
5	0.918	0.902	1.7
6	0.952	0.937	1.5
7	0.97	0.957	1.2
8	0.977	0.969	0.8
9	0.978	0.974	0.4
10	0.979	0.978	0.1

Table 7: B=(4,4,3,3,6,2), μ=(1,3,2,4,3,4)

# cust	exact throughput	approximate throughput	relative error %
3	0.71	0.702	1.15
4	0.798	0.787	1.33
5	0.841	0.831	1.12
6	0.853	0.848	0.5

Table 8: B=(2,2,2,2,2,2,2), μ=(1,3,2,4,3,4,3)

# cust	exact throughput	approximate throughput	relative error %
3	0.685	0.679	0.95
4	0.769	0.763	0.79
5	0.811	0.807	0.47
6	0.826	0.824	0.34

Table 9: B=(2,2,2,2,2,2,2), μ=(2,1,4,5,2,3,3)

# cust	exact throughput	approximate throughput	relative error %
3	0.889	0.898	1.0
4	1.051	1.068	1.6
5	1.168	1.197	2.4
6	1.236	1.243	0.6
7	1.268	1.269	0.1

Table 10: B=(2,2,2,2,2,2,2,2), μ=(5,3,4,6,2,3,3,2)

# cust	exact throughput	approximate throughput	relative error %
4	1.379	1.391	0.9
5	1.477	1.506	1.94
6	1.541	1.57	1.8
7	1.583	1.593	0.65
9	1.606	1.597	0.5
10	1.587	1.584	0.2
11	1.549	1.559	0.4
12	1.487	1.495	0.57
13	1.385	1.394	0.6

Table 11: B=(6,2,2,4) , μ=(3,2,4,2)

# cust	exact throughput	approximate throughput	relative error %
4	0.889	0.892	0.3
5	0.935	0.94	0.5
6	0.961	0.965	0.5
7	0.973	0.975	0.2
9	0.978	0.977	0.1
10	0.974	0.972	0.2
11	0.964	0.962	0.2
12	0.936	0.94	0.4
13	0.881	0.899	2

Table 12: B=(3,4,5,2), μ=(2,1,4,2)

# cust	exact throughput	approximate throughput	relative error %
4	0.849	0.843	0.74
5	0.895	0.888	0.79
6	0.917	0.914	0.3
7	0.926	0.925	0.2
8	0.93	0.93	0
10	0.931	0.931	0
11	0.929	0.928	0.1
12	0.923	0.924	0.1
13	0.909	0.912	0.4
14	0.87	0.892	2.4

Table 13: B=(4.3.2,4,2), μ=(3.2,4.2.1)

# cust	exact throughput	approximate throughput	relative error %
4	0.59	0.583	1.2
5	0.637	0.63	1.1
6	0.66	0.656	0.6
7	0.668	0.665	0.4
9	0.668	0.667	0.1
10	0.6598	0.6597	0
11	0.629	0.633	0.6

Table 14: B=(2.2.2.2,2.2), μ=(1.1,1.3.2.3)

# cust	exact throughput	approximate throughput	relative error %
4	0.699	0.704	0.8
5	0.755	0.767	1.65
6	0.793	0.803	1.2
7	0.817	0.816	0.2
9	0.804	0.815	1.3
10	0.773	0.8	3.4
11	0.722	0.738	2.2

Table 15: B=(2.2.2.2.2.2), μ=(2,1,4,3,1,4)

# cust	exact throughput	approximate throughput	relative error %
4	0.796	0.787	1.1
5	0.86	0.847	1.5
6	0.893	0.885	0.9
7	0.913	0.907	0.66
8	0.92	0.917	0.5
9	0.925	0.922	0.3
11	0.918	0.921	0.3
12	0.902	0.913	1.2
13	0.859	0.882	2.8

Table 16: B=(2,2,2.2.2.2.2), μ=(3,2.4,5,1.2,3)

# cust	exact throughput	approximate throughput	relative error %
4	1.033	1.043	0.97
5	1.157	1.183	2.2
6	1.252	1.297	3.6
7	1.327	1.39	4.7
8	1.389	1.426	2.7
9	1.434	1.449	1
10	1.46	1.462	0.14
12	1.457	1.461	0.3
13	1.427	1.444	1.2
14	1.381	1.415	2.5
15	1.309	1.317	0.6
16	1.186	1.173	1.1

Table 17: B=(3,2,3.3.2.2.2), μ=(4,2.2,3,5.2,3)

# cust	exact throughput	approximate throughput	relative error %
4	0.649	0.648	0.15
5	0.712	0.713	0.14
6	0.758	0.761	0.4
7	0.79	0.795	0.63
8	0.811	0.811	0
9	0.823	0.821	0.24
10	0.827	0.826	0.12
12	0.826	0.825	0.12
13	0.816	0.818	0.25
14	0.795	0.805	1.3
15	0.762	0.769	1.1
16	0.704	0.714	1.4

Table 18:B=(3,2.3,3,2,2.2), μ=(3.1,2.1.2,3,4)

#cust	exact throughput	approximate throughput	relative error%
4	1.738	1.738	0
5	1.818	1.813	0.25
6	1.866	1.853	0.7
7	1.884	1.874	0.5
8	1.889	1.884	0.3
9	1.891	1.889	0.1
10	1.892	1.891	0.04

Table 19:B=(∞,2,3,5),μ=(8,2,3,2),p_{1i}=(0.0.25,0.25,0.5)

#cust	exact throughput	approximate throughput	relative error%
4	1.827	1.83	0.2
5	1.882	1.884	0.04
6	1.914	1.908	0.3
7	1.923	1.919	0.2
8	1.926	1.924	0.02
9	1.927	1.926	0.05
10	1.927	1.927	0

Table 20:B=(∞,2,3,5),μ=(18,2.3,2),p_{1i}=(0,0.25,0.25.0.5)

#cust	exact throughput	approximate throughput	relative error%
4	1.945	1.94	0.3
5	1.979	1.972	0.3
6	1.989	1.985	0.2
7	1.992	1.99	0.1
8	1.99296	1.99197	0.05
9	1.99311	1.99302	0
10	1.99314	1.99302	0
11	1.99315	1.99314	0
12	1.99317	1.99316	0
13	1.99319	1.9932	0

Table 21:B=(∞,3,4,6),μ=(4,6,3,8),p_{1i}=(0,0.25,0.25,0.5)

#cust	exact throughput	approximate throughput	relative error%
4	1.188	1.179	0.7
5	1.223	1.214	0.8
6	1.236	1.229	0.6
7	1.239	1.236	0.3
8	1.24	1.239	0.1
9	1.241	1.24	0.05
10	1.241	1.241	0

Table 22:B=(∞.2,3,5).μ=(4,3,3,4),p_{1i}=(0.1/3.1/3.1/3)

#cust	exact throughput	approximate throughput	relative error%
4	1.798	1.738	0.6
5	1.357	1.345	0.7
6	1.38	1.37	0.5
7	1.388	1.383	0.3
8	1.39	1.388	0.1
9	1.3907	1.39	0.05
10	1.3908	1.3906	0

Table 23:$B=(\infty,2,3.5),\mu=(4.3,2.5),p_{1i}=(0.0.3.0.2.0.5)$

#cust	exact throughput	approximate throughput	relative error%
4	1.216	1.212	0.4
5	1.244	1.239	0.4
6	1.252	1.249	0.3
7	1.254	1.253	0.1
8	1.2545	1.254	0.04
9	1.25453	1.2543	0

Table 24:$B=(\infty,2,3,4),\mu=(4,3,3,4),p_{1i}=(0,0.45,0.2,0.35)$

#cust	exact throughput	approximate throughput	relative error%
5	0.62	0.616	0.6
6	0.651	0.644	1.1
7	0.669	0.661	1.2
8	0.679	0.672	0.98
9	0.683	0.678	0.66
10	0.684	0.682	0.4
11	0.68467	0.6835	0.2
12	0.68474	0.6844	0.05

Table 25:$B=(\infty,3,5,4),\mu=(3,1,1,1),p_{1i}=(0,1/3,1/3,1/3)$

#cust	exact throughput	approximate throughput	relative error%
4	1.944	1.938	0.32
5	1.976	1.969	0.36
6	1.985	1.981	0.22
7	1.988	1.986	0.11
8	1.9883	1.9874	0.05
9	1.9884	1.9881	0.02
10	1.98844	1.98832	0
11	1.98844	1.98841	0
12	1.98844	1.98843	0

Table 26:$B=(\infty,3,4,6),\mu=(4,6,3,8),p_{1i}=(0,1/4,1/4,1/2)$

#cust	exact throughput	approximate throughput	relative error%
3	1.649	1.638	0.72
4	1.77	1.75	1.13
5	1.819	1.803	0.89
6	1.837	1.827	0.53
7	1.8421	1.8371	0.27
8	1.8433	1.8412	0.12
9	1.8435	1.8427	0.04

Table 27:$B=(\infty,2,3,5),\mu=(4,3,2,5),p_{1i}=(0,0.3,1/5,1/2)$

#cust	exact throughput	approximate throughput	relative error%
3	1.589	1.604	0.96
4	1.727	1.738	0.62
5	1.805	1.808	0.2
6	1.848	1.843	0.3
7	1.863	1.857	0.3
8	1.8665	1.3631	0.18
9	1.8673	1.3658	0.08

Table 28:$B=(\infty,2,3,5),\mu=(8,2,3,2),p_{1i}=(0,1/4,1/4,1/2)$

#cust	exact throughput	approximate throughput	relative error%
4	0.57	0.583	2.21
5	0.615	0.622	1.01
6	0.643	0.645	0.27
7	0.658	0.658	0
8	0.665	0.664	0.14
9	0.6673	0.6661	0.17
10	0.6681	0.6672	0.14
11	0.6682	0.6677	0.08

Table 29:$B=(\infty,3,5,4),\mu=(3,1,1,1),p_{1i}=(0,1/3,1/3,1/3)$

#cust	exact throughput	approximate throughput	relative error%
3	1.088	1.076	1.11
4	1.163	1.146	1.41
5	1.191	1.179	1
6	1.1994	1.1932	0.52
7	1.202	1.199	0.25
8	1.2025	1.2013	0.1
9	1.2027	1.2022	0.04

Table 30:$B=(\infty,2,3,5),\mu=(4,3.3.4),p_{1i}=(0,1/3,1/3,1/3)$

#cust	exact throughput	approximate throughput	relative error%
3	1.116	1.103	1.2
4	1.174	1.161	1.1
5	1.193	1.185	0.66
6	1.1979	1.194	0.32
7	1.1989	1.1974	0.12
8	1.19901	1.1986	0.03

Table 31:$B=(\infty,2,3,4),\mu=(4,3,3,4),p_{1i}=(0,0.45,0.2,0.35)$

#cust	exact throughput	approximate throughput	relative error%
4	0.714	0.718	0.57
5	0.759	0.754	0.61
6	0.777	0.772	0.63
7	0.784	0.781	0.42
8	0.7856	0.7836	0.25
9	0.7859	0.785	0.11

Table 32:$B=(\infty,3,4,3),\mu=(3,2,1,2),p_{1i}=(0,1/3,1/3,1/3)$