Queueing Network Analysis Tool (QNAT):
A GUI Based Queueing Network Analysis Tool

Hema Tahilramani[*], ECSE Dept, Rensselaer Polytechnic Inst, Troy NY
D. Manjunath, Dept of Elecl Engg, IIT,Bombay Powai Mumbai INDIA
Sanjay K. Bose, Dept of Elecl Engg, IIT,Kanpur Kanpur INDIA [†]
email: `tahilh@rpi.edu,dmanju@ee.iitb.ernet.in,skb@iitk.ac.in`

Abstract

QNAT is a powerful GUI based Mathematica library for the Microsoft Windows environment that can be used to analyze a large class of queueing networks. Arbitrary configurations of multiserver queues with infinite or finite capacity, fork-join queues with or without synchronization queues can be specified through a GUI. Multiple classes of jobs can be specified and each job class can be independently a closed class or an open class. If there are finite capacity queues in the system, the type of blocking - transfer or repetitive service, can also be specified. The latest version is v2.0, a 32 bit version and works on Mathematica version 3.0 in Windows95, Windows 98 or Windows NT environments. An evaluation copy is available from the authors.

[*]Corresponding author

[†]Other contributors to the development of QNAT are D. M. Bhaskar, currently with Deptt of Telecommunications, Govt of India, Madras, India and M. N. Umesh, currently with Silicon Automation Systems, Bangalore, India.

1 Introduction

Queueing network models have been extensively used for performance analysis of telecommunication systems, manufacturing systems, computer systems, vehicular systems, etc. Although realistic representations of these systems are very complex and tractable analytical models not readily available, analytic expressions for the performance parameters using simplified models are preferrred over simulations models because of the higher efficiency of the former. A tool that can be used to quickly and easily describe the queueing system and solve the system and have the performance parameters conveniently available is expected to be very useful for the performance analyst of the above mentioned systems. In this paper we describe one such tool that has been developed in the Electrical Engineering Department of IIT,Kanpur. The first version of this tool has been distributed over the Internet and has found many users.

Software to describe and analyse queueing systems have been available. We are aware of the existence of CADS (1982) BEST/1 (1982), Capture/MVS (1982) ResQ (1982), MAP (1982) [3], PANACEA (1982), and QNA [1, 2]. There are many drawbacks with these tools. Firstly, these were not easily availble to us. Secondly, in these tools, the system description is through a text file which, in these days of GUI based user friendly computing would not be very convenient. Thirdly, most of these packages were capable of solving only a restricted class of queueing networks and their analysis capabilities were limited by the approximate and exact analysis techniques available at that time. With advances in computational platforms and in the development of the theory of queueing networks to solve many new classes of systems, a unified tool, powerful enough to analyze a wide range of queueing networks and provide accurate results becomes necessary. To allow a quick ramp up on the learning curve a GUI based, user-friendly front end becomes imperative. QNAT is an effort to provide such a tool on the Windows platform for the performance analyst. In this paper we describe the features, capabilities and the analysis techniques used in QNAT. In the next section we provide a brief overview of QNAT and in Sections 3 and 4 we describe the analysis techniques that we use to solve the various models that QNAT can solve. We summarise the features of QNAT in Section 5.

2 Overview of QNAT

QNAT is a GUI based tool for analysing queueing networks that works on the Windows platform and uses the symbolic computing software, Mathematica, as the computational platform. The operations corresponding Mathematica are hidden from the user. Thus it is necessary to have Mathematica to be able to run QNAT. The first option is to decide if the network will have finite capacity nodes, *i.e.*, if the network will be blocking. This choice is essential because the analysis techniques to analyse such networks is considerably different from that of networks without finite capacity queues. The following kinds of queueing networks can be analysed using QNAT.

- Networks with only infinte capacity nodes:
  In this category, the network could have
  - A single class of jobs in multi server queues with infinite capacity queues. The network could be open or closed and fork-join nodes can also be present in the system. (Fork-join nodes are very useful in modeling manufacturing flow lines and in constructing models of distributed computing systems.). The synchronising queue in the fork-join node may or may not be present.
  - Multiple classes of jobs each of which could be open or closed. Fork-join nodes that handle multiple classes of jobs are not allowed for such networks.
- Networks in which some or all queues have a finite capacity and can cause blocking of jobs. For such networks, at this time we allow only a single class of jobs in the networks. The network could be open or closed and the blocking mechanisms for the network that can be specified are transfer-blocking, repetitive-service-blocking with fixed or random destination and rejection-blocking.

To describe a new network, a `Start` option on the menu bar is used. On selecting this option, the following choices need to be made to help QNAT to internally decide the solution algorithm that it will use for this

- Number of classes of jobs in the network
- Type of network - open, closed or mixed for non blocking networks and kind of blocking for blocking networks.
- Number of nodes in the network.
- Number of customers in each closed class

The above defines the network and a `Create` option draws the network on the screen. Each of the nodes may now be visited through the GUI to define the parameters of the node through pop up menus. The number of servers in the queue, the service discipline, parameters for the external arrivals to the node, service time parameters, for each class of jobs at the node, routing probabilities for jobs finishing service at the node, if the node is a fork-join node, the parameters for such a system are some of the description items of the nodes. For fork-join nodes, the number of sibling queues and the presence or absence of synchronising queues needs to be specified. All the sibling queues in a fork-join node are assumed to be single server queues.

A network can be saved into a file and then loaded again at a later time. The network description is saved as an ASCII file using a storage convention internal to QNAT.

After describing the network, (either a creating a new network or loading a previously stored network) analysis is initiated by choosing a `Run` option. This command processes the network and launches Mathematica. Once the analysis is completed, the performance parameters for each node is obtained through the GUI by visiting the node and selecting an `Output` option. Network level results are also available. The results can be printed using a `Print Run Results` option which will save a text file with the user-specified name containing a description of the network and the results of the analysis.

Online help is provided with QNAT. At this time, this help is not extensive but is aimed at providing soem clarifications in situations where some additional information or assistance is considered necessary.

3 Analysis of Non Blocking Networks

For networks of infinite capacity queues, QNAT may be used to analyze open, closed and mixed networks with one or more classes of jobs. The analytical methods also support various types of fork-join queueingmodels for the individual nodes.

If the network is open with only one class of jobs, the techniques described in [1] are followed. This allows some generalisation in the description of the arrival and service processes at the individual nodes in the network by allowing the specification of the two moments of the interarrival and service times. If the squared coefficient of variation (SqV), defined as the ratio of variance to the square of the mean of the inter arrival times and the service times is unity, the resulting network will be a Jackson network and the solution is exact, otherwise the solution is approximate and the quality of the approximation is described in [2]. If there are multiple classes of jobs, then the MVA technique is used in the analysis.

3.1 Open Networks

In an open queueing network, jobs arrive into the network from outside the network, receive service at one of mutiple queues and finally depart from the network. Assuming that arrival rate into the system is known, the goal in analysis of open queuing networks is to characterize the distribution of number of jobs in the system. To analyse these networks, we use the GI/G/m method outlined in [1, 2]. This is a two-moment approximation technique and can handle fixed or probabilistic routing for single class of jobs. The first two moments of the interarrival time and the service time distributions are specified as input. The typical output parameters are the first two moments of the distributions of the number of jobs and the delays at all the individual nodes in the network. Server utilization at each queue and the first two moments of the sojourn time and total number in the network are other performance measures that are available through QNAT. For analytical convenience, the second moment is expressed as the squared coeeficient of variaton, SqV. We will use this description for the second moment wherever we need to use second moments.

The first and second moments of the net arrival process at each node in the network may be obtained by solving a set of linear flow balance equations. The performance of the network is then obtained by analyzing each queue in isolation. Each queue is a standard GI/G/m queue characterized by first two moments of the arrival process and the service times. At each node i, a multiplication factor γ_i is used to specify the creation or combination of jobs at that node. If $\gamma_i > 1$, then jobs are created, if $\gamma_i < 1$, then jobs are combined and if $\gamma_i = 1$, then jobs are neither created nor combined. If λ_i is the net arrival rate into node i, then departure rate out of node i will be $\lambda_i \gamma_i$.

It will be assumed that all the queues in the network have infinite buffer capacity and that the service discipline at all the nodes is FCFS.

3.2 Closed Networks

In a closed network there are no external arrivals to the system and hence no departures out of the system. We use the well known Mean Value Analysis (MVA) technique to evaluate the performance of closed queueing networks. The performance parameters that will be computed are the mean queue length, the mean waiting time, mean throughput etc. The well known convolution technique can be used to find higher moments of the performance parameters but we choose to use the MVA technique because of its numerical stability and popularity.

The MVA technique is based on the well known Mean Value Theorem [3, 4]. To use this technique to analyse closed networks, we need to define the concept of visit ratio of each job class to each node which is a measure of the relatve arrival rate to a node of jobs from each class. Thus, visit ratio at node k, V_k is the ratio of throughput of node k, to the throughput of the reference node in the network and node 1 is arbitrarily selected as the reference node in QNAT to calculate the visit ratios. It is easy to see that $V_k{=}\lambda_k/\lambda_1$ where λ_k is the absolute arrival rate to node k.

Multiple classes of jobs can be specified. It will be assumed that the service times are all exponentially distributed at all the nodes for all the classes and the service discipline at each node is either of FCFS, LCFS with preemptive resume, processor sharing (PS) or infinite server (IS) queue.

3.3 Mixed Networks

In mixed networks there are multiple classes of jobs with some open classes and some closed classes of jobs. The open classes in the network are characterised by the external arrival rates for that class and the closed classes are characterised by the population of that class. In QNAT we use the algorithm described in [3] to solve mixed queueing networks. Since this is not very widely used, we briefly describe this algorithm.

The utilization of the open classes at each node is first computed and then the net utilisation at each node due to the open classes is computed by summing the individual utilisation. This method is similar to the solution of open networks. The performance parameters of the closed classes are then computed by eliminating the open classes and converting the mixed model into a closed model with inflated service demands for the closed classes. The inflation factor used is $1 - U_{i(O)i}$, where $U_{i,(O)}$ is the utilisation due to the open classes at node i and is given by

$$U_{i,(O)} = \sum_c U_{ik}$$

Here, U_{ik} is the utilisation at node i for class k jobs. This is the percentage of time the server is not used by open classes and is the time available for the closed classes to get service from the server in the queue. The performance of the closed classes is analysed using the standard MVA solution technique discussed earlier with the inflated service times for the closed class jobs. The performance of the open classes are computed by considering the mean queue length of the closed classes at each node in the network. For these type of networks, QNAT permits the use of zero open class with non zero closed classes or vice-versa. Thus the analysis of pure closed and pure open networks is a special case of the analysis of mixed networks.

3.4 Analysis of Fork Join Nodes

In an open network in QNAT, a fork-join node may specified only without synchronising queues at the output of the sibling queues. When the synchronising queues are not specified, the sub jobs that finish early will wait at the server till all the sub jobs corresponding to this job have finished service and block that server. Thus the effective service time distribution of a job at the fork-join node will be the distribution of the maximum of k iid service times. The fork-join node is reduced to a "normal node" by specifying the mean and SqV of this distribution for the normal node. After reduction to a normal node, the open network is solved as before. If the network is closed, the reduction of this node to a "normal node" is by first obtaining the distribution of the maximum of k service time distributions and fitting a minimum mean squared error (MMSE) exponential distribution to it. The mean of this distribution is then used in MVA for analyzing the overall network.

In QNAT fork-join queues with synchronizing queues can be specified only for closed classes. If synchronising queues are specified, sub-jobs wait in the synchronising queue after they complete service at a sibling queue, *i.e.*, the server at the sibling queue is freed after service to service another job that may have forked. When all the subjobs finish their service, they are recombined and exit the node as one job. The fork-join node is replaced by a Flow Equivalent Server (FES) that has a state dependent service rate. Although an exact FES can be obtained using a brute force technique, it is computationally very intensive and is used only when there are a small number of siblings. When the number of siblings is large, an approximation technique based on decomposition and aggregation bastechnique outlined in [5, 6] is used. In this method, a two sibling system is solved exactly and its FES obtained. This is used with a third sibling to solve the system exactly again and the process repeated till the FES representation of the $k-$sibling system is obtained. The FES replaces the fork join node and the system solved using the well known MVA technique.

In the specifucation of a fork join node, it is assumed that external arrivals to the sibling queues are not allowed and that jobs cannot be created or combined at these queues. All the sibling queues will be assumed to be single server queues with an exponential service time distribution. Further, it will be assumed that only one class of jobs will be served at a fork-join node.

4 Analysis of Blocking Networks

In QNAT, networks with finite capacity queues can also be specified and analysed. The theory for such networks that allow multiple classes of customers has not been well developed and we allow only a single class of jobs in such networks. It should be noted that it is not necessary to that all the nodes in the network be of finite capacity.

Depending on the what is done following a job finishing service at a node being blocked, three types blocking networks are known - transfer blocking, repetitive service blocking and rejection blocking. In transfer blocking, the blocked job waits at the server from where it has just finished service and blocks it till the destination becomes unblocked. In repetitive service blocking, the blocked job receives another service at the server from where it has finished service. At the end of this service, it could choose the same destination or a different destination. In rejection blocking, a blocked job leaves the network. Obviously, rejection blocking cannot be specified for a closed network. For an open network it is assumed that under all blocking mechanisms, an external arrival that is blocked is lost. In the following we briefly outline the analysis technique we use in QNAT for blocking networks. In some of these networks, generalised exponential (GE) distributions for the interarrival and service times are also permitted.

4.1 Open Networks with Transfer Blocking

We use the technique developed by us and outlined in [9]. In this technique, the interarrival and service times could have GE distributions and the individual nodes could be GE/GE/m/N queues. This is an iterative technique similar to the analysis of GI/G/m queues by Whitt [1] and uses flow balance equations to determine the arrival and service process at the nodes and Maximum Entropy techniques of [8] to obtain the marginal distributions of the number of jobs at each node. The iterative process is continued till the performance parameters of interest converge. It will be assumed that the service discipline is FCFS. Some or all the nodes in the network may have infinite capacity. Since our interest is in a performnace model, we will assume that in a dealdock situation, the network immediately resolves the deadlock condition by simultaneously moving all the blocked jobs to their destinations. This scheme is also described in [13].

4.2 Open Networks with Repetitive Service Blocking

As described earlier, in this type of blocking a blocked job receives multiple services at the source node. In repetitive service blocking with fixed destination (RS-FD) the job attempts to join the same destination node after subsequent service completions. In repetitive service blocking with random destinations (RS-FD), the destination node is chosen according to the routing probabilities after every service completion. In QNAT , we have used the Maximum Entropy Method described in [8] for the analysis of arbitrary open queueing networks with multiple general servers and Repetitive Service blocking. This method is outlined below.

Consider an arbitrary open queueing network under RS-RD or RS-FD blocking mechanisms. This network consists of N FCFS multiple server queues of finite or infinite capacity. Let $\mathbf{n} = [n_1, n_2, \cdots, n_N]$ be the state of the system where n_i is the number of jobs in node i. It has been shown in [8] that under steady state, the Maximum Entropy (ME) solution, for the system state, $p(\mathbf{n})$, subject to normalization and certain other constraints, is given by the product-form approximation

$$p(\mathbf{n}) = \prod_{i=1}^{N} p_i(n_i)$$

where, $p_i(n_i)$ is the marginal ME solution of the i^{th} node which is a GE/GE/m_i/N_i queue. (The Maximum Entropy technique is based on the principle that "of all the distributions satisfying the set of constraints imposed by the system, the most likely distribution is the one that maximises the system's entropy".) Thus an open network is decomposed into individual GE/GE/m/N queues with revised inter-arrival and service time distributions. All the incoming streams (including the external arrival stream) to node i merge to form the overall arrival stream to node i. The effective service time at node i is the total time for which a server of queue i is occupied by a particular job.

4.3 Open Networks with Rejection Blocking

In rejection blocking, a job arriving to a full node is lost, *i.e.*, forced to leave the network. In QNAT , to analyse these networks we use the method suggested in [7].

This technique is based on the approximation that the net flow into a node, consisting of the external arrivals and the arrivals from the other nodes in the network, forms a Poisson process. Flow balance equations are solved iteratively in order to find the net offered traffic to each node in the network. The equations are not the same as those for Jackson type networks, but are similar and are modified to incorporate rejection blocking. A product form approximation for the steady state probabilities is made with the marginal distributions of the state at each queue given by the steady state distribution of M/M/m/K queues.

In the analysis of these networks, apart from the usual performance parameters like average number in the queues and average sojourn times, parameters like conditional expected sojourn time for the jobs that are lost due to blocking (U_L) and the conditional expected sojourn time for the jobs that leave the network after completing all their service requirements (U_s) will also be important. These parameters, U_s and U_L, are calculated by considering the passage of a job through the queueing network as a random walk with "Loss" and "Served" as the absorbing states. These states represent the situations when a job was lost on encountering a full queue or when the job left the network after completing its service requirements. Jobs that are lost before starting service at any node in the network are called rejected jobs. Lost jobs also include the rejected jobs.

4.4 Closed Networks with Repetitive Service Blocking

In QNAT, the Maximum Entropy Method [8] is used for the analysis closed networks with repetitive service blocking. Arbitrary topologies with multiple servers at the nodes can be specified. The technique that we use is the one developed in [8] and its brief description follows. Corresponding to the closed network specified, a "pseudo-open" network is defined. The pseudo open network is an equivalent open network with no external arrival streams and no external departures and satisfying the principle of "conservation of population" which is represented by the fixed mean population constraint that the sum of the mean number of jobs at all the nodes in network must be equal to the population size of the closed network.

The fixed mean population constraint poses a restriction on the minimum number of jobs that may be present in a queue of the pseudo open network. If L is the total number of jobs in a network, then the minimum number of jobs at any node i will be givn by $K_i = \max\{0, L - \sum_{i \neq j} N_i\}$ where, N_i is the capacity of node i. In closed networks, K_i will be greater than zero for some node i if the total number of jobs circulating in the network is greater than the total capacity of the network excluding node i. If L is the number of jobs in the network, a node with capacity greater than or equal to L will effectively behave as an infinite capacity node. To address this, the pseudo open network is decomposed into individual *censored* queues with revised inter arrival and service time distributions and solved in the same way as an open network under repetitive service blocking. (For more on censored queues, see [8].) The solution obtained from the pseudo open network may not satisfy the job flow balance equations. The values of the flow balance coefficients are then iteratively adjusted so that the flow balance equations are satisfied. The marginal state probabilities obtained after these iterations are used to obtain the performance measures of interest.

To describe closed queueing networks with repetitive service blocking in QNAT generalised exponential (GE) ditributions are permitted for the inter arrival and service times.

4.5 Closed Network with Transfer Blocking

The product-form approximation method of [12] is used in QNAT to analyze a closed network with transfer blocking. In this method, an identical network with same parameters but nodes having infinite capacity is first considered. Note that all the states in the equivalent non blocking network will not be feasible in the blocking network. The states which violate the station capacities are *normalized* as follows. If the number of jobs i n a node exceeds its capacity, then the number in the node is set equal to the capacity of the node and the excess jobs are distributed among all the other nodes depending on the incoming transition probabilities (from the rest of the nodes to the node whose capacity constraints were being violated). This is continued till a feasible state is obtained. A feasible state is a state valid for the blocking network. By normalizing all the infeasible states of the identical non blocking network, we obtain an equivalent state space which consists only of those states which are feasible for the blocking network. The probability $p(\mathbf{n})$ of a feasible state $\mathbf{n}$ is the sum of the probabilities of all the states of the equivalent non blocking network which map into this feasible state. Since the equivalent non blocking network has a product-form solution, Jackson's results may be used to calculate the state probabilities. Based on the normalization described above, the state distribution of the blocking network may be found from that of the non blocking network by *normalization*. All the performance parameters, except throughput, can now be calculated from this state distribution. Calculation of the throughput is based on the fact that a non blocking network with an equal number of states as the blocking network has the same stochastic structure as that of the blocking network. The throughput of these networks will also be approximately the same. In order to compute the throughput, the number of states in the blocking queueing network is determined. The equivalent number of jobs in the non blocking network is then obtained such that the two networks have same number of states. The non blocking queueing network with this equivalent number of jobs is now analysed to obtain its throughput and this is used as the approximate throughput of the blocking network.

In the analysis of these networks, it is assumed that in situations where more than one customer is blocked by the same node, the blocked customers enter the node on a first-blocked-first-enter basis.

5 Summary and Conclusions

In this paper we have described QNAT, a Microsoft Windows based software tool for queueing network analysis. This package may be used to analyse general queueing networks with nodes having finite or infinite buffer capacity. QNAT has a user-friendly GUI based front end to define the network and to view the results of the solution. Mathematica has been used as the computational platform which simplifies the development process and enabled the easy implementation of mathematically involved algorithms. Another salient feature of QNAT is that the analysis of many types of queueing networks can be done using in one package. Depending on the specification of the network, QNAT uses the right set of packages/algorithms to carry out the analysis.

An earlier version of QNAT has been distributed over the Internet for comments, evaluation and debugging. For an evaluiation copy of the latest version of QNAT , the authors may be contacted. A web site containing the latest information on QNAT at `http://www.iitk.ernet.in/pub2/ee/networks/QNAT/` may also be accessed.

A Input and Output Parameters to QNAT

In the following we list the input parameters that will describe the network, and the output parameters that will be available from QNAT for each category of queueing network that can be specified.

Open Non Blocking Networks

Input Parameters

$\mathcal{N}$	Number of nodes in the network
m_i	Number of servers at node i
λ_{oi}	External arrival rate at node i
C_{oi}	SqV of external interarrival times to node i
$1/\mu_i$	Mean of service time distribution at node i
CS_i	SqV the service time distribution at node i
γ_i	Multiplication factor at node i
$\mathbf{P} = \{p_{ij}\}$	Routing matrix

Output Parameters for each Node

λ_i	Mean arrival rate at node i
C_{ai}	SqV of interarrival times into node i
EN_i	Mean of the number of jobs at i
CN_i	SqV of the distribution of the number of jobs at node i
EW_i	Mean of waiting time at node i
CW_i	SqV of waiting time at node i
Dep_i	Departure rate from node i
V_i	Visit counts at node i
U_i	Utilisation at node i
NW	Mean sojourn time through the network.
NN	Mean number of jobs in the network
NCN	SqV of distribution of number of jobs in network

Closed Non Blocking Networks

Input Parameters

$\mathcal{N}$	Number of nodes in the network
R	Number of classes in the network
m_i	Number of servers at node node i
$1/\mu_{ik}$	Mean service time at node i for jobs of class k
$Qtype_i$	Service discipline at node i
$\mathbf{C}$	Population vector $= \{C_1, ..C_R\}$ where C_k is population of class k
$[\mathbf{P}_k]$	$\equiv \{q_{k(i,j)}\}$, Routing matrix for class k.

Output parameters

EN_{ik}	Mean number of class k jobs at node i
EW_{ik}	Mean waiting time for class k jobs at node i
λ_{ik}	Mean throughput of class k jobs at node i
V_{ik}	Visit ratio of class k jobs at node i
EN_i	$EN_i = \sum_{k=1}^{R} EN_{ik}$, mean number of jobs at node i

Mixed Networks

Input parameters

Ro	Number of open classes in the network
Rc	Number of closed classes in the network
R	Total number of classes in the network, $R = Ro + Rc$
$\mathcal{N}$	Number of nodes in the network
m_i	Number of servers at node i
$[\mathbf{C}]$	Population vector for closed classes, $[\mathbf{C}] = \{C_1..C_{Rc}\}$
$[\mathbf{P}_k]$	Routing matrix for class k
$\lambda_{O,ik}$	External arrival rate of open class k jobs at node i
$1/\mu_{ik}$	Mean service time for class k jobs at node i
γ_{ik}	Multiplication factor for jobs of open class k at node i

Output parameters

EN_o	mean open class jobs at node i
EN_c	mean closed class jobs at node i
λik	net arrival of class k at the node i
EN_{ik}	Mean number of jobs of class k at node i
EW_{ik}	Mean delay for class k jobs at node i
Dep_{ik}	Departure rate of class k jobs at node i
V_{ik}	Visit count of open class k or visit ratio of closed class k at node i
U_{ik}	Utilization of class k jobs at node i

Open Blocking Networks

Input Parameters

$\mathcal{N}$	Number of nodes in the network
m_i	Number of homogeneous servers at node i
M_i	Capacity of node i, including number of servers
$1/\mu_i$	¡ean service time of a server at node i
C_{si}	SqV of the service time at node i
$[\mathbf{P}] = \{p_{ij}\}$	Routing Matrix
λ_{oi}	External arrival rate to node i
C_{oi}	SqV of the external interarrival times at node i

Output Parameters

EW_i	Average delay at node i
EN_i	Average number of jobs at node i
γ_i	Throughput from node i
U_i	Utilization of the servers at node i
PB_f^i	Probability that a job completing service at node i is blocked
PB_{ij}	Probability that a job completing service at node i is blocked by node j
γ	Network throughput
S	Mean sojourn time through the network

Closed Blocking Networks

Input Parameters

$\mathcal{N}$	Number of nodes in the network
K	Number of jobs in the network
m_i	Number of servers at node i
N_i	Capacity of node i (including the number of servers)
$1/\mu_i$	Mean service time of a server at node i
L	Number of jobs in the network
C_{si}	SqV of the service time at node i
$[\mathbf{P}] = \{p_{ij}\}$	Routing Matrix

Output Parameters

V_i	Visit ratio of jobs to node i with respect to node 1
D_i	Average delay at node i
K_i	Mean number of jobs at node i
S_i	Throughput at node i
U_i	Utilization of node i
PB_{di}	Blocking probability for job completing service at node i
PB_{ij}	Blocking probability for job completing service at node i, going to node j
SN	Network throughput
DN	Average delay between arrival and departire of jobs to the network

References

[1] W. Whitt, "The Queueing Network Analyser," *Bell System Technical Journal*, vol 62, no 9, Nov 1983, pp 2779-2815.

[2] W. Whitt, "Performance of the Queueing Network Analyser," *Bell System Technical Journal*, vol 62, no 9, Nov 1983, pp 2817-2843.

[3] Edward D. Lazowska, John Zahorjan, G. Scott Graham & Kenneth C. Sevcik, *Quantitative System Performance*, Prentice-Hall Inc., 1984.

[4] Michael K. Molloy,*Fundamental of Performance Modelling*, Macmillan Publishing Company, 1989.

[5] Y. C. Liu and H. G. Perros, "A Decomposition for the of a Closed Fork/Join Queueing System," *IEEE Trans on Computers*, vol 40, no 3, pp 365-370, March 1991.

[6] Y. C. Liu and H. G. Perros, "Approximate Analysis of a Closed Fork/Join model," *Euro Jour of Oper Res* vol 53, pp 382-392, 1991.

[7] O. Bronstein and I. B. Gertsbakh, "An Open Exponential Queueing Network with Limited Waiting Spaces and Losses: A Method of Approximate Analysis," *Performance Evaluation,* vol 4, 1984, pp 31-43.

[8] D. D. Kouvastos and N. P. Xenios, "MEM for Aribitrary Queueing Networks with Multiple General Servers and Repetitive Service Blocking," *Performance Evaluation,* vol 10, 1989, pp 169-195.

[9] H. Tahilramani, *Analysis of Various Types of Blocking in Finite Capacity Queueing Networks,* M. Tech Thesis, Dept. of Elecl Engg, Indian Inst of Tech,Kanpur, India, Feb 1998.

[10] L. Kerbache and J. MacGregor Smith, "The Generalized Expansion Method for Open Finite Queueing Networks," *Euro Jour of Oper Res,* vol 32, 1987, pp 448-461.

[11] Hyo-Seong Lee and S. M. Pollock, "Approximate Analysis of Open Acyclic Exponential Queueing Networks with Blocking," *Operations Research,* vol 38, 1990, pp 1123-1134.

[12] I. F. Akyildiz, "Product Form Approximations for Queueing Networks with Multiple Servers and Blocking," *IEEE Trans on Computers,* vol 38, no 1, 1989, pp 99-113.

[13] H. Tahilramani, D. Manjunath and Sanjay K. Bose, "Approximate Analysis of Open Network of GE/GE/m/N Queues with Transfer Blocking," *submitted to MASCOTS 99.*