PERFORMANCE ANALYSIS OF NETWORKS OF OPEN ZERO-BUFFER MULTI-SERVER QUEUES

Frederico R. B. Cruz (UFMG)
fcruz@est.ufmg.br

In this paper we aim to accurately evaluate the performance of open zero-buffer multi-server general queueing networks. The measure of interest is the throughput, which is evaluated by mean of a well-know tool, the Generalized Expansion Metthod (GEM). The GEM is a node-by-node decomposition method successfully used in the past to approximately evaluate the performance of finite queues. We compare the results provided by the GEM with those of simulation. Our experiments attest for the quality of the GEM. A wide range of testing instances was analyzed, including different basic topologies. For all cases tested, the errors were below 16%, which is quite satisfactory for optimization purposes, to be carried out in the next step of this research.

Palavras-chaves: Finite queues, bufferless queues, performance evaluation.

1. Introduction and Motivation

Networks of finite queues without buffers occur in many real-life physical systems in the semi-process and process industries (for more information, see FRANSOO & RUTTEN, 1994). A zero-buffer production environment might be necessary due to the processing technology of the product itself or simply due to the absence of any intermediate storage capacity between two consecutive operations of a job.

A steel production process is described by Hall and Sriskandarajah (1996). Molten steel goes through a series of operations ranging from ingots, un-molding, reheating, soaking, and preliminary rolling. In this production process, the steel must pass one operation to another continuously, without any waiting or buffering of work-in-process, since such waiting would result in cooling down the steel to a temperature that is not acceptable for the next process. Hence, either a job is finished and transferred directly to the next process, or it is buffered in the machine itself until the downstream process is ready to receive another job. Similarly, in food-processing environments no buffer space is allowed between the cooking operation and before the canning operation. This is due to the requirement that the product should still be fresh when it is canned. Similar issues can be found in producing juice and beer. In these cited examples, restrictions in the processing technology and its characteristics create zero-buffer production system.

The nature of the product dictates hygienic consideration as one of the critical factor in production of condiments such as mayonnaise and various types of salad dressing as the product. As studied by Ramudhin and Ratliff (1995), there is no space for work-in-process inventory and the product must never wait between two operations. As a last example, third generation mobile communication networks are characterized by a multi-server zero-buffer queueing system (TSYBAKOV, 2002). In such systems, arrivals are represented by requests of audio, data, and video messages, whereas the service time is the message transmission time. Here, zero-buffers are caused due to simple absence of storage capacity between operations. Despite the high industrial relevance of zero-buffer networks particularly in process and semi-process industries, only scant literature is available focusing exclusively on these types of networks.

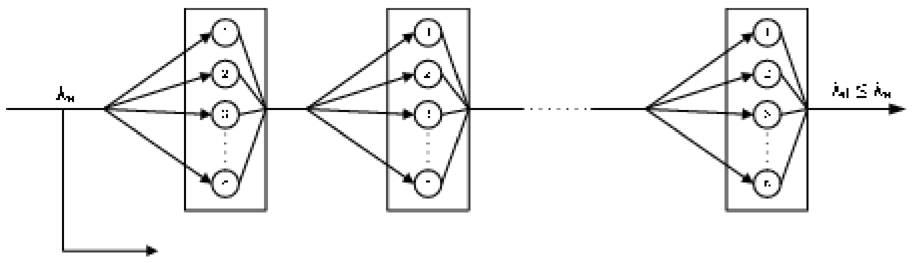


Figure 1: A serial zero-buffer queuing network

We are interested in zero-buffer multi-server general queueing networks, seen in Figure 1, as a queueing network representation for a tandem line. Zero-buffer systems are a special case of a restricted queueing network. Restricted queueing networks have a finite capacity in each node, referred to as the total buffer capacity of size K_j. That is, a finite node j can only hold entities up to a certain quantity K_j including those entities in service. The buffer capacity at finite node j causes blocking to occur when the arriving quantity to node j exceeds its buffer capacity K_j (BUZACOTT & SHANTHIKUMAR, 1993). As a consequence, each node in the network might be affected by events at other nodes, leading to the phenomena of blocking and starvation (PERROS, 1994).

A particular case where a queueing network has a finite capacity but no buffers before servers is denoted as a zero-buffer queueing network (also denoted in the literature as bufferless, no intermediate buffers etc.). In this specific case, the buffer space at node *j* is equal to the number of server c_j in that node, that is, $K_j = c_j$. Given that there is no space to queue, a job in the upstream node can only enter the downstream node if the servers have finished processing their jobs (see Figure 1).

In the remaining of this paper, we present briefly the method used for approximate performance evaluation of open zero-buffer multi-server queueing networks. Following, we compare the approximations with simulation, for a number of different topologies and setting. Final remarks and topics for future research in the area conclude the paper.

2. Performance Evaluation Algorithm

The GEM is a robust and effective approximation technique developed by Kerbache and Smith (1987). It has been successfully used to estimate performance measures for finite queueing networks. As described in previous papers, this method is basically a combination of repeated trials and node-by-node decomposition in which each queue is analyzed separately and then corrections are made in order to take into account the interrelation between the queues in the network. The GEM uses the blocking after service (BBS) protocol, which is prevalent in most production and manufacturing, transportation, and other similar systems. In this section, we present an overview of the method. For more detailed information and applications of the GEM, the reader is referred to the papers by Kerbache and Smith (1987, 1988, 2000), Jain and Smith (1994), Spinellis et al. (2000), and Smith and Cruz (2005).

The GEM involves three stages, namely, network reconfiguration, parameter estimation, and feedback elimination. We shall describe them briefly as follows.

2.1 Network reconfiguration

The first step in the GEM includes reconfiguring the network by adding an artificial queue for each queue that is succeeded by a finite queue, in order to register the blocked entities, as seen in Figure 2.

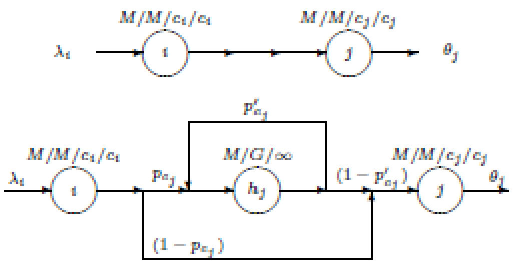


Figure 2: The generalized expansion method (GEM)

2.2 Parameter estimation

In the second stage, the estimation of three important parameters is carried out. The first one is the blocking probability of the downstream nodes *j*, given by the well-known Erlang loss formula (for clarity, we omit the subscripts)

$$p_c = \frac{(\lambda/\mu)^c / c!}{\sum_{i=0}^{c} (\lambda/\mu)^c / c!},$$

that is, for systems under Markovian arrivals, general services, *c* parallel servers, and the total capacity of *c* users, including those in service.

The second parameter is the probability that an entity is forced back to the holding node h_j, given that it was rejected at the previous trial, p_c', which may be approximated by a method from Labetoulle and Pujolle (1980).

Finally, the third parameter, the service rate for the holding node, may be given by renewal theory (KLEINROCK, 1975) as

$$\mu_h = \frac{2\mu_j}{1+\sigma_j^2 \mu_j^2}.$$

2.3 Feedback elimination

The repeated visits to the holding node (due to the feedbacks) create dependences in the arrival process. In order to eliminate the immediate feedback, the entities are given an extra service time during the first passage through the holding node and the adapted service rate then becomes

$$\mu_h' = (1 - p_c')\mu_h.$$

2.4 Summary

In summary, the GEM ultimate goal is to provide an approximation scheme to update the service rates of the upstream nodes, taking into account the blocking after service that happens there, caused by the downstream nodes, that is,

$$\tilde{\mu}_i^{-1} = \mu_i^{-1} + p_{c_j} (\mu_{h_j}')^{-1}.$$

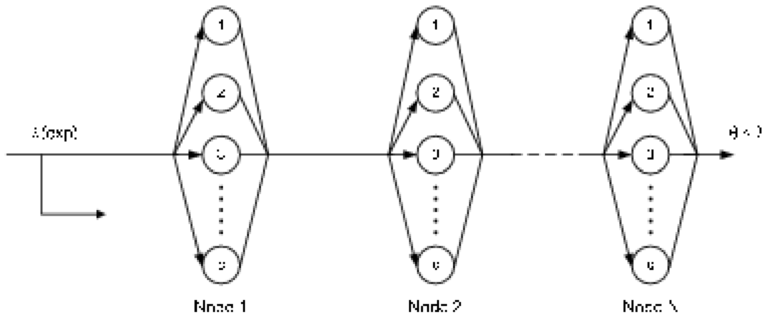
The throughput at a node *i*, succeeded by a finite node *j*, is then obtained as follows (subscripts were omitted for the sake of clarity),

$$\theta = \lambda(1 - p_c) = \lambda \left(1 - \frac{(\lambda/\tilde{\mu})^c / c!}{\sum_{i=0}^{c} (\lambda/\tilde{\mu})^c / c!} \right),$$

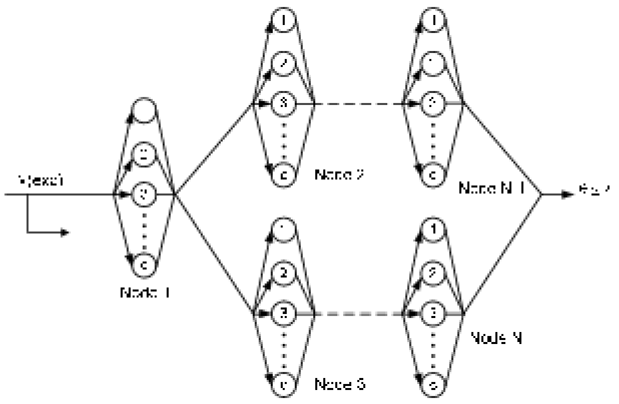
and the overall throughput is obtained simply by adding up the throughput at the last(s) node(s) of the networks.

3. Computational Experiments

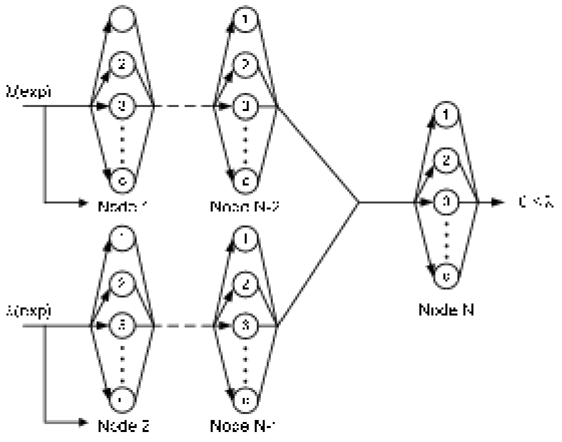
In order to evaluate the quality of the approximations given by the GEM, for zero-buffer queueing networks, experiments were conducted using three different topologies, namely series, split, and merge. These topologies were examined in symmetrical and asymmetrical settings. The configurations are seen in Figure 2.



a) series topology



b) split topology



c) merge topology

Figure 2: Topologies examined.

For each topology, several number of nodes, servers, and arrival rates were tested, $N \in \{3, 5, 9\}$, $c \in \{2, 4, 10\}$, and $\lambda \in \{2, 4, 8, 16\}$, respectively. All servers were considered with service rate $\mu = 10$. We combined the above parameters in 36 experiments, as seen in Table 1. In the split topology, Figure 2-b, the arriving jobs were divided into two streams departing from the fist node in the system. The splitting node is always positioned directly after the first node. The routing probabilities for both routes are set to be equal to each other (0.5). In the merge topology, Figure 2-c, the jobs arrive from two different source nodes. The overall arrival rate is then divided equally for the two source nodes. The last node merges the two streams.

Topology	N	**c**	λ	θ	θ_s	δ	CPU(mim)	$\Delta\%\theta$
Series	3	(2,2,2)	2	1.996	1.966	0.001	2.0	1.53
			4	3.949	3.952	0.002	5.8	-0.09
			8	7.394	7.424	0.003	12	-0.41
			16	11.50	11.32	0.002	22	1.59
	5	(4,4,4,4,4)	2	2.000	1.999	0.002	5.0	0.05
			4	4.000	3.999	0.002	5.2	0.03
			8	7.988	7.987	0.003	20	0.02
			16	15.61	15.61	0.004	40	0.03
	9	(10,10,…,10)	2	2.000	2.000	0.001	9.0	0.02
			4	4.000	4.000	0.002	18	0.00
			8	8.000	8.002	0.003	36	-0.02
			16	16.00	16.00	0.003	73	-0.01
Merge	3	(2,2,2)	2	2.000	1.991	0.002	1.9	0.45
			4	3.993	3.930	0.001	3.8	1.61
			8	7.900	7.473	0.002	8.2	5.71
			16	14.52	12.54	0.003	14	15.80
	5	(4,4,4,4,4)	2	2.000	1.999	0.001	3.0	0.04
			4	4.000	3.999	0.002	5.8	0.03
			8	7.999	7.993	0.003	12	0.08
			16	15.98	15.88	0.003	24	0.63
	9	(10,10,...,10)	2	2.000	1.999	0.001	5.3	0.04
			4	4.000	4.00	0.003	10	0.05
			8	8.000	7.998	0.003	20	0.03
			16	16.00	16.00	0.005	45	0.00
Split	3	(2,2,2)	2	1.997	1.967	0.002	2.1	1.53
			4	3.957	3.783	0.002	4.8	4.59
			8	7.538	6.785	0.002	8.9	11.1
			16	12.59	10.65	0.002	15	18.2
	5	(4,4,4,4,4)	2	2.000	2.000	0.001	4.5	0.00
			4	4.000	3.996	0.002	5.7	0.10
			8	7.988	7.936	0.003	17	0.65
			16	15.65	15.09	0.004	28	3.69
	9	(10,...,10)	2	2.000	2.000	0.002	12	0.00
			4	4.000	3.999	0.002	13	0.02
			8	8.000	7.999	0.003	25	0.01
			16	16.00	16.00	0.004	150	0.00

Table 1: Results for the symmetrical queueing networks

In order to attest for the quality of the solutions given by the GEM, simulation experiments were set up. The simulations were conducted using ARENA (KELTON ET AL., 2001). We used an observation time of 200,000 time units and a warm-up period (see details in ROBINSON, 2007) of 2,000 time units, for 20 independent replications. Then, we compute the % deviation for the analytical results of the throughput, defined as $\Delta\%\theta = 100\%(\theta-\theta_s)/\theta_s$, in which θ is the throughput given by the GEM and θ_s is the (somewhat) exact throughput given by the simulation. The results are shown in Table 1, in which we see that the analytical results, although not always as accurate as desirable, are mostly reasonable and acceptable. In other words, under extreme high utilization rates, that is, heavy traffic and quite few servers, the error may be as high as 16-19%, but in the majority of the cases, the error is 6% or better (low). Table 1 also shows the half-width of the 95% confidence interval given by the simulations (column δ) and the total cpu time, in minutes, for performing each one of the simulations.

Topology	c	μ	λ	θ	θ_s	δ	CPU(mim)	Δ%θ
Series	(2,4,10)	(12,11,10)	2	1.998	1.977	0.001	3.0	1.06
			4	3.974	3.973	0.002	6.3	0.02
			8	7.698	7.698	0.002	12	0.00
			16	13.50	13.50	0.002	43	0.00
	(2,4,10,2,4)	(12,11,10,12,11)	2	1.998	1.997	0.001	5.0	0.04
			4	3.974	3.839	0.002	9.9	3.52
			8	7.697	7.700	0.003	20	-0.04
			16	13.47	13.50	0.002	37	-0.21
	(2,4,10,2,4,10,2,4,10)	(12,11,10,12,11,10,12,11,10)	2	1.998	1.999	0.001	11	-0.03
			4	3.974	3.973	0.002	18	0.03
			8	7.969	7.759	0.002	35	2.71
			16	13.45	13.355	0.003	68	0.71
Merge	(4,4,2)	(11,11,12)	2	2.000	2.000	0.002	1.9	-0.01
			4	4.000	4.000	0.002	4.2	0.00
			8	8.000	7.987	0.003	8.2	0.16
			16	15.92	15.472	0.003	17	2.90
	(10,10,4,4,2)	(10,10,11,11,12)	2	2.000	2.000	0.002	3.0	-0.01
			4	4.000	3.999	0.002	5.8	0.03
			8	8.000	7.998	0.002	12	0.03
			16	15.93	16.00	0.005	29	-0.44
	(2,2,4,4,10,10,4,4,2)	(12,12,11,11,10,10,11,11,12)	2	2.000	1.994	0.001	5.1	0.33
			4	3.996	3.952	0.002	10.1	1.12
			8	7.947	7.652	0.003	20	3.86
			16	15.35	14.12	0.002	41	8.73
Split	(2,4,4)	(12,11,11)	2	1.998	1.973	0.001	2.1	1.27
			4	3.974	3.839	0.002	2.5	3.51
			8	7.698	7.057	0.003	4.7	9.08
			16	13.51	11.585	0.003	8.1	16.6
	(2,4,4,10,10)	(12,11,11,10,10)	2	1.998	1.977	0.002	1.8	1.09
			4	3.974	3.980	0.002	6.1	-0.15
			8	7.698	7.058	0.002	6.7	9.07
			16	13.51	11.585	0.003	11	16.6
	(2,4,4,10,10,4,4,2,2)	(12,11,11,10,10,11,11,12,12)	2	1.998	1.976	0.001	3.0	1.12
			4	3.974	3.839	0.002	5.8	3.51
			8	7.698	7.654	0.002	22	0.57
			16	13.51	12.88	0.030	38	4.89

Table 2: Results for the asymmetrical cases

We also considered asymmetrical cases, with unbalanced settings for the routing probabilities in the split topologies (routing probabilities 0.4-0.6, in the splitting nodes, Figure 2-b) and for the arrival rates in the merge topologies (external arrivals 0.4λ and 0.6λ, in the front nodes, Figure 2-c). Also, different service rates and different number of servers were assumed along the networks, as seen Table 2, columns ***c*** and μ. From the results seen in Table 2, we observe that the main conclusions drawn earlier hold. That is, the errors may be considerably high under high utilization. We can see that the blocking effects become more important with the increase of the arrival rates.

As a final word concerning the simulations, we note that the GEM tends to overestimate the throughput but also to underestimate it sometimes, when compared to the results from the simulations (as reflected by the values seen in the column of the Δ%θ), exactly as one should expect when comparing simulation with unbiased analytical results.

4. Conclusions and Final Remarks

The Generalized Expansion Method (GEM) was used here as an approximate performance evaluation tool for finite zero-buffer queueing networks. We have shown that the method typically delivers results within 5% of error, for basic series, merges, and split topologies, and for both symmetrical and asymmetrical setting. The maximum error observed, although, may be considerably higher, around 20%, mainly for those configurations under very heavy traffic. These are new results as the GEM has not been used before to specifically evaluate the networks of open zero-buffer multi-server queus presented here. The GEM is fast (typically runs in a split second) and may provide a relatively simple tool to evaluate the throughput rate, both in low and moderate blocking probability settings, which may be useful for optimization purposes. In fact, the application of the GEM in an optimization framework should be the subject of future papers.

References

BUZACOTT, J. & SHANTHIKUMAR, J. G. *Stochastic Models of Manufacturing Systems*, Prentice-Hall; 1993.

FRANSOO J. C. & RUTTEN W. G. M. M. *A typology of production control situations in process industries*. International Journal of Operations and Production Management. Vol. 14, n. 12, p. 47-57, 1994.

HALL NG, SRISKANDARAJAH C. *A survey of machine scheduling problems with blocking and no-wait in process*. Operations Research. Vol. 44, p. 510-525, 1996.

JAIN S. & SMITH, J. M. *Open finite queueing networks with M/M/C/K parallel servers*. Computers & Operations Research. Vol. 21, n. 3, p. 297-317, 1994.

KELTON, D.; SADOWSKI, R. P. & SADOWSKI, D.A. *Simulation with Arena*, McGraw Hill College Div., New York; 2001.

KERBACHE, L. & SMITH, J. M. *Assymptotic behavior of the expansion method for open finite queueing networks*. Computers & Operations Research. Vol. 15 , n. 2, p. 157-169, 1988.

KERBACHE, L. & SMITH, J. M. Multi-objective routing within large scale facilities using open finite queueing networks. European Journal of Operational Research. Vol.121, p. 105-123, 2000.

KERBACHE, L. & SMITH, J. M. *The generalized expansion method for open finite queueing networks*. European Journal of Operational Research. Vol 32, p. 448-461, 1987.

KLEINROCK, L. *Queueing Systems*, Vol. I: Theory, John Wiley & Sons, New York; 1975.

LABETOULLE, J. & PUJOLLE, G. *Isolation method in a network of queues*. IEEE Transactions on Software Engineering. Vol. SE-6, n. 4, p. 373-381, 1980.

PERROS, H. G. *Queueing Networks with Blocking*, Oxford University Press; 1994.

RAMUDHIN A. & RATLIFF, H. D. *Generating daily production schedules in process industries*. IIE Transactions. Vol. 27, p. 646-656, 1995.

ROBINSON, S. *A statistical process control approach to selecting a warm-up period for a discrete-event simulation*. European Journal of Operational Research. Vol. 176, n. 1, p. 332-346, 2007.

SMITH, J. M. & CRUZ, F. R. B. *The buffer allocation problem for general finite buffer queueing networks*. IIE Transactions. Vol. 37, n. 4, p. 343-365, 2005.

SPINELLIS, D.; PAPODOPOULOS, C. & SMITH, J. M. *Large production line optimisation using simulated* *annealing*. International Journal of Production Research. Vol 38, n. 3, p. 509-541, 2000.

TSYBAKOV, B. *Optimum discarding in a bufferless system*. Queueing Systems. Vol. 41, p. 165-197, 2002.