Research Paper

Volume : 3 | Issue : 1 | January 2014 • ISSN No 2277 - 8179

Merge Configuration of Multiservers with Finite Capacity Intermediate Buffers and External Arrivals to the Merger Queue

Computer Science

KEYWORDS : queueing network, merging, blocking after service

Angel Vassilev Nikolov | Department of Mathematics and Computer Science, National University of Lesotho, NUL 180 Roma, Lesotho

ABSTRACT

We consider a queueing network consisting of multiservers in parallel and connected to a merger queue. All servers have a finite capacity buffers. Some incoming streams go directly to the merger without passing through merging servers. We insert a virtual servers for these requests and decompose the configuration into two subsystems: merging multiservers and merger multiserver, and then analyze the subsystems in isolation. Decomposition and aggregation are used and performance metrics including the occupancy probabilities are obtained.

1. Introduction

Queueing network models are extensively used to describe the behavior of complex real-life physical systems in manufacturing and production, health care,, computer and software engineering, networking, see e.g. [2, 7, 8, 9, 11].

The methods applied fall into two major categories – simulation and analytical. The simulation is most sophisticated and detailed representation of the system and produces most accurate results related to performance [9]. It is most expensive, however, in terms of development and computing time, and prone to design errors because of its complexity.

The analytical models are simpler and less expensive and in many cases, especially at early design stages, provide good estimates of the performance metrics. One of the most common analytical approaches is based on the principle of decomposition and aggregation. It is implemented in three steps:

1) network decomposition into subsystems,
2) analysis of each subsystem in isolation,
3) analysis of the new aggregated system.

In [8] the network is broken down to nodes (single servers or multiservers) and a system of equations for the underlying Markov process is developed. The equations for all nodes are to be solved simultaneously. The states of each node are the numbers of active, waiting and blocked units, respectively, and this gives rise to the number of the unknowns.

Another heuristic model is developed in [10]. Introduction of pseudo-arrival rates and effective service rates makes possible analysis of broad range of network types where blocked arrivals are rejected.

The occupancy probabilities for the aggregated merger server of merge configuration with single servers and finite buffers can be computed from a simple birth-death process, and then performance metrics can be expressed as a function of these probabilities, thus significantly reducing the number of the unknowns [5, 6]. In [7] a merge configuration of multiservers and single merger is analyzed.

In this paper we develop model of merge configuration with multiserver as a merger. Some incoming requests are directly connected to the merged queue and others pass through multiservers before joining the merger.

2. Model Description and Analysis

The system studied consists of merged queue, referred below as a server *0* or queue *0* as well connected to *M* Poisson input streams. Part of these arrivals originates from multiservers[i], and others are directly connected to the outside world. We shall refer to the streams directly connected to the queue *0* as external arrivals. Such case is very common in network traffic.[2,4]. Arrivals to the merged queue are independent. In order to develop an appropriate model we insert a virtual server with zero service time between the incoming

stream and its input to the queue *0*. This virtual server does not change the operation of the system. From the point of view of the merger, however, there are *M* servers (real and virtual) connected to it, and this a merge configuration. No constraints are imposed on the merged server, it can also consist of multiple units. The index *i* can be viewed as the sequential number of the servers and runs from *1* to M unless otherwise stated. The arrivals to any multiserver are dispatched to some of its units if there is one free, and if all single units are occupied the request waits in the corresponding buffer. When an arrival finds no free position in the buffer (buffer is full), however, it is lost, or rejected.

After completion of the service at server *i* the request proceeds to multiserver *0*, and is taken immediately into service if there is a free unit (no queue at server *0*), otherwise is placed into the buffer *0* and waits. Requests at queue *0* are served on First Come First Served (FCFS) basis until there is at least one free position in buffer *0*. When buffer *0* becomes full all incoming arrivals from are blocked, and the merging servers are forced to wait, so that there is no loss of requests at server *0*, but some delay is inserted due to the fact the this blocking after service (BAS) leads to a closing of a single unit from the multiserver from which the request originates. The number of the operating units of that multiserver is reduced by one. As the time progresses the number of units *Li* forced to shut down can vary from *0* to *Ni*, i.e. at some point all units of multiserver *i* are blocked and it produces no requests to server *0*. For virtual servers is assumed *Ni* =1in order to be treated as the other merging servers. Blocking contributes to the degradation of the total system throughput, i.e. to the increase of the number of the rejected requests at the inputs of the real and virtual merging servers by inserting additional time required to wait to join the merged queue. If *j* units are blocked by the merged queue then any newly generated request from the merging queues must wait for the server *0* to complete the service of *j+1* requests, or in other words to make free *j+1* places in its buffer.

The approach used is to decompose the system into individual queues, and analyze each individual queue separately. We assume that there are Poisson arrivals with rate $\tilde{\lambda}_i^{N_i-L_i}$ from merging server *i* to server *0* as long as $N_i - L_i$ of its units are not blocked. Obviously arrival rates decrease as the number of the blocked unit *Li* increases, and at some point $\tilde{\lambda}_i^0 = 0$ that is merging server *i* becomes fully detached. Since blocked units are forced to wait at queue *0* the size of the buffer of queue *0* can be regarded as extended by $K=\sum_{i=1}^{M} N_i$ so its total size is $N + K$.

We introduce the following notations:

B_i size of the buffer for server *i*

N_i number of units for multiserver *i*

c number of units for server *0*

N size of the buffer for server *0*

$T_i^{N_i-L_i}$ clearance time of multiserver *i* given that $N_i - L_i$ units are not blocked. This is the time between entrance of service at queue *i* and arrival at queue *0*. If there are no blocked units at the merger queue $T_i^{N_i-L_i}$ is equal to the service time of the unit $1/\mu_i$. If, however, *j* units in the system are blocked the time needed to complete the service of *j+ 1* requests by server *0* must be added to $1/\mu_i$ as explained above. Its distribution represents the sum of *j +1* independent

Volume : 3 | Issue : 1 | January 2014 • ISSN No 2277 - 8179

exponentially distributed random variables, each of which has a parameter of μ.

$f_i^{N_i-L_i}$ P [i^{th} queue is full given $N_i - L_i$ units are operational (not blocked)]

$\bar{\lambda}_i^{N_i-L_i}$ arrival rate to merging server *i* as long as $N_i - L_i$ units are operational (not blocked). Apparently rejected (lost) requests are excluded from this rate.

$\tilde{\lambda}_i^{N_i-L_i}$ arrival rate to the merger queue from multiserver *i* as long as $N_i - L_i$ units are operational (not blocked)

$b_i^{N_i-L_i}[n]$ blocking probability; P[more than L_i units of multiserver *i* are blocked and the number of requests (waiting +blocked units) at queue *0* is *N* + *n*]

$\alpha_i^{N_i-L_i}[N+j]$ conditional probability that, upon completion of service at server *i* when $N_i - L_i$ units are operational and there are *N*+ *j* units at queue *0*, a request is blocked. It can be viewed as well as a probability that at the point of completion of service at multiserver *i*, the new arrival sees only $L_i$units blocked, there are *N* + *j* units in the extended buffer of server *0*, given that $N_i - L_i$ units are not blocked.

f_i P[queue *i* is full]; probability of rejection the incoming request at queue *i*

$\bar{\lambda}_i = \lambda_i(1 - f_i)$ throughput for queue *i*

$\bar{\lambda} = \sum_{i=1}^{M} \bar{\lambda}_i$ total system throughput

$K = \sum_{i=1}^{M} N_i$

$\delta_{m,n}$– Kronecker delta symbol

$\prod_n^m = 1$ if *n*>*m*

$\sum_n^m = 0$, if *n*>*m*

The system is considered to be balanced and all probabilities in this paper are steady-state ones.

2.1. Analysis of the Merged Queue

The goal of this analysis is to express the occupancy probabilities of queue *0* as a function of the arrival rates $\tilde{\lambda}_i^{N_i-L_i}$, N_i, and *N*. The arrival rates $\tilde{\lambda}_i^{N_i-L_i}$ are measured on the output of the multiserver *i*. The states of the queue are represented by the number of the requests in the queue, but if there is a blocking the order of the blocked units counts, since they are being unblocked on a FCFS basis, and consequently the number of the states increases enormously. The problem is dealt with disregarding the order of the blocked units, and aggregating the states, which have same number of units into one state. The states of such aggregated system are denoted by *k (k = 0, ...,N + K)* and represent the number of units waiting for service at queue *0* including the blocked units, if any. We define $\lambda_a(k)$ as a transition rate from state *k* to state *k + 1* for the aggregated system. After solving the aggregated queue the occupancy probabilities inclusive of the order of arrivals of the blocked units are expressed in terms of the aggregated probabilities. The aggregated states of queue *0* follow simple birth and death process and its steady-state probabilities are determined by the following expressions

$$P[k] = P[0] \prod_{j=1}^{k} \frac{\lambda_a/\mu}{k!} \quad k=1,...,c \qquad (1)$$

$$P[k] = P[0] \prod_{j=1}^{k} \frac{\lambda_a/\mu}{c^{k-c}c!} \quad k=c+1,...,N+K \qquad (2)$$

$$\sum_{k=0}^{N+K} P[k] = 1 \qquad (3)$$

By $L_n^p = [L_1, .., L_{i,...,} L_M]$ we denote an integer *1xM* vector for which $0 \le L_1 \le N_1$, …, $0 \le L_i \le N_{i,}$ …, $0 \le L_M \le N_M$, and $\sum_{i=1}^{M} L_i = n$ hold. Superscript *p* points to a particular vector from the set of such vectors. We also define a *1xn* vector $z_n^{p,q} = [z_1, \dots, z_l, \dots z_n]$ $(z_l \in \{1, \dots, i, .. M\})$ for which $\sum_{l=1}^{n} \delta_{i,z_l} = L_i$ holds for all element of the vector L_n^p. *q* is a sequential number runs from *1* to $\prod_{i=1}^{M} \binom{n-\sum_{l=1}^{i-1} L_l}{L_i}$ where term $\prod_{i=1}^{M} \binom{n-\sum_{l=1}^{i-1} L_l}{L_i}$ is the number of ways $L_1, .., L_{i,...,} L_M$ distinct elements can be placed in *n* distinct places.. In the context of the nonaggregated queue *0* each vector $z_n^{p,q}$ represents a unique order of arrivals from multiservers *i* where the numbers of units from each multiserver are uniquely determined by the vector L_n^p. For example if M = 3, n = 4 and $L_4^p = [2, 0, 2]$ the corresponding vectors $z_4^{p,q}$ are $z_4^{p,1}$=[1, 1, 3, 3], $z_4^{p,2}$=[[1, 3, 1, 3], $z_4^{p,3}$=[1, 3, 3, 1], $z_4^{p,4}$=[3, 3, 1, 1], $z_4^{p,5}$=[3, 1, 3, 1], and $z_4^{p,6}$=[3,1, 1, 3].The occupancy probabilities of the original queue *0* are denoted as *P[k]* for *k = 1, ...,N* and $P[N+n; z_n^{p,q}]$ for *n = 1, ...,K*. We prove the following theorem:

*Theorem 1:*The relationship between the probabilities of the original queue 0 and those of the aggregated queue is:

$$P[N+n; z_1, \dots, z_l, \dots, z_n] = P[N+n] \frac{\prod_{i=1}^{M} \prod_{k=N_i-L_i+1}^{N_i} \tilde{\lambda}_i^k}{\Omega_n} \quad (4)$$

where

$$\lambda_a(k) = \sum_{i=1}^{M} \tilde{\lambda}_i^{N_i} \text{ for } k=1,\dots,c+N \quad (5)$$

$$\lambda_a(N+n) = \frac{\Omega_{n+1}}{\Omega_n}$$

for *n=1,...,K-1* (6)

$$\Omega_n = \sum_{\forall L_n^p} \prod_{i=1}^{M} \binom{n-\sum_{l=1}^{i-1} L_l}{L_i} \prod_{k=N_i-L_i+1}^{N_i} \tilde{\lambda}_i^k \quad (7)$$

For external arrivals $\tilde{\lambda}_i^1 = \lambda_i$.

Proof:

Proof is done by substitution of (4) in the balance equations of the nonaggregated queue *0,* similarly to [5, 7]

2.2. Analysis of the Merging Queues

In this subsection we first express the rates $\tilde{\lambda}_i^{N_i-L_i}$ in terms of the parameters of the multiservers. Applying the conservation flow rule to queue *i* yields

$$\bar{\lambda}_i^{N_i-L_i} = \tilde{\lambda}_i^{N_i-L_i} (\delta_{L_i,0} \sum_{k=0}^{N} P[k] + \sum_{\forall (z_n^{p,q} \& L_n^p(i)=L_i} P[N+n; z_1, ., z_l, ., z_n])$$

$$L_i = 0, \dots, N_i - 1 \quad (8)$$

The term in parenthesis above is the probability that there are $N_i - L_i$ nonblocked units of multiserver *i.* The rate $\bar{\lambda}_i^{N_i-L_i}$ is measured at the input of queue *0* and is smaller than $\tilde{\lambda}_i^{N_i-L_i}$ because of the blocking.

From the definition of $b_i^{N_i-L_i}[n]$ follows:

$$b_i^{N_i-L_i}[0] = 0 \quad (9)$$

$$b_i^{N_i-L_i}[n] = P[c+N+n].$$

Volume : 3 | Issue : 1 | January 2014 • ISSN No 2277 - 8179

$$\sum_{\forall(z_n^{p,q}\&L_n^p(i)>L_i)} \frac{\prod_{i=1}^{M}\prod_{k=N_i-L_i+1}^{N_i}\tilde{\lambda}_i^k}{\Omega_n}$$

for $L_i = 0, \dots, N_i - 1$ (10)

Conditional probabilities $\alpha_i^{N_i-L_i}[c+N+j]$ are:

$$\alpha_i^{N_i-L_i}[c+N+j] = \frac{P[N+j]-b_i^{N_i-L_i}[j]}{1-\sum_{n=1}^{K} b_i^{N_i-L_i}[n]}$$

i=1,...,M; j=0,...,K

(11)

and the expected values of the clearance times $E[T_i^{N_i-L_i}]$:

$$E\left[T_i^{N_i-L_i}\right] = \frac{1}{\mu_i}$$

$$+\sum_{j=0}^{K-1}\alpha_i^{N_i-L_i}[c+N+j]\frac{1}{c\mu} \quad j=0,\dots,K$$

(12)

If queue *0* is connected to other queues the term $\frac{1}{\mu}$ in the above expression should be replaced by the expected value of the clearance time of the downstream $E[T_D]$.

We denote $\rho_i^{N_i-L_i} = \lambda_i\, E\left[T_i^{N_i-L_i}\right]$

and derive the probabilities of rejection at multiserver *i*

$$f_i^{N_i-L_i} = P_i^{N_i-L_i}[0]\frac{(\rho_i^{N_i-L_i})^{B_i+N_i-L_i}}{(N_i-L_i)!(N_i-L_i)^{B_i}} \quad (13)$$

where

$$P_i^{N_i-L_i}[0] = 1/\left(\sum_{k=0}^{N_i-L_i}\frac{(\rho_i^{N_i-L_i})^k}{k!}\right.$$

$$\left.+\frac{(\rho_i^{N_i-L_i})^{(N_i-L_i)}}{(N_i-L_i)!}\sum_{k=N_i-L_i+1}^{B_i+N_i-L_i}\frac{(\rho_i^{N_i-L_i})^k}{k!}\right)$$

(14)

$$f_i = f_i^{N_i-L_i}(\delta_{L_i,0}\sum_{k=0}^{N}P[k]+$$

$$\sum_{\forall(z_n^{p,q}\&L_n^p(i)=L_i} P[c+N+n; z_1,.,z_l,.,z_n])$$

(15)

For external arrivals:

$$f_i = P[\text{merger is full}]$$

$$= 1-\sum_{n=0}^{K}P[c+N+n] \quad (16)$$

Solving equations (1-16) simultaneously produces the wanted performance metrics and occupancy probabilities.

In the case of asynchronous web server after completion of service at queue *0* the job is routed to a server for specific processing tied to the upstream. [8]. The routing probability equals obviously $\bar{\lambda}_i/\bar{\lambda}$ and is not fixed as many analyzes suggest.

3. Concluding Remarks

The merits of this model could be summarized as follows:
1) Relatively small number of equations.
2) Deep insight into the performance metrics since all probabilities, including those of all permutations of the blocked units can be calculated with insignificant computational effort.
3) Applicable to multiclass job arrivals.

REFERENCE

[1] I.A. Akyyildiz, H.V. Brand, Exact solutions for networks of queues with | blocking-after-service, Theoretical Comp. Sci., 125 (1994), 111-130. | [2] S. Balsamo, V. De N. Persone, P. Inverardi, A review on queueing network | models with finite capacity queues for software architectures performance | prediction, Perf. Eval., 974 (2002), 1-20. | [3] U.N. Bhat, Introduction to Queuing Theory: Modelling and Analysis in | Applications, Springer (2008). | APPROXIMATE MODEL FOR A MERGE CONFIGURATION... 401 | [4] H. Chen, H.Q. Ye, Methods of diffusion approximation for multi-server systems: | Sandwich, uniform attraction and state-space collapse, In: Queueing | Networks: a Fundamental Approach, Springer (2011), 488-530. | [5] H.S. Lee, S.M. Pollock, Approximate analysis for the merge configuration | of an open queueing network with blocking, Technical Report, University | of Michigan, 87-11 (1987), 1-28. | [6] H.S. Lee, S.M. Pollock, Approximate analysis of open exponential queueing | networks with blocking: General configuration, Technical Report, Univer- | sity of Michigan, 48-109 (1987), 1-31. | [7] C. Osorio, M. Bierlaire, A finite capacity queueing network model capturing | blocking, congestion and spillbacks, European J. Operational Res., 196 | (2009), 996-1007. | [8] U. Praphamontripong, S. Gokhale, A. Gokhale, J. Gray, Performance analysis | of an asynchronous web server, In: Comp. Software and Appl. Conf., | 2006 COMPSAC'06 30th Ann. Intnl., 2 (2006). | [9] R. Suri, S. Sahu, M. Vernon, Approximate mean value analysis for closed | queueing networks with multiple-server stations, In: Proc. of the 2007 Ind. | Eng. Res. Conf., Nashwille (2007). | [10] Y. Takahachi, H. Miyahara, T. Hasegawa, An approximation method for | open restricted queueing networks, Operational Res., 28 (1980), 594-602. | [11] Y. L. Zhu, S. Y. Zhu, H. Xiong, Performance analysis and testing of the | storage area network, In: Proc. of the 10th NASA Goddard Conf. on Mass | Storage Systems and Technologies 2002, Singapore (2002). | 402 |