Asian Journal of Information Technology 13 (3): 126-130, 2014
ISSN: 1682-3915

Analytical Model for Performance Evaluation of an Asynchronous Web Server

Angel V. Nikolov
Department of Mathematics and Computer Science,
National University of Lesotho, Roma, Lesotho

Abstract: This study deals with analytical performance evaluation of asynchronous web server. The queuing model of the server consists of merge configuration of multiple servers and splitting configuration of single servers in tandem. A set of few equations is presented solution of which produces performance metrics including the occupancy probabilities of all servers. Numerical results show significant improvement of the performance as the number of the multiserver's units increases.

Key words: Asynchronous web server, merging, splitting, BAS, tandem

INTRODUCTION

Queueing network models are extensively used to describe the behavior of complex computer and software architectures (Balsamo *et al*., 2002; Cao *et al*., 2003; Gokhale *et al*., 2005; Praphamontripong *et al*., 2006). 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. It is most expensive, however, in terms of development and computing time and prone to design errors because of its complexity (Praphamontripong *et al*., 2006).

The analytical models are simpler and less expensive and in many cases, especially at early design stages, provide good estimates of the performance metrics. In Cao *et al*. (2003), a M/G/1/K*PS queuing model of a synchronous web server is presented. In Gokhale *et al*. (2005), a model of the reactor pattern is introduced based on Stochastic Reward net modeling paradigm.

Approximate analytical approaches give relatively accurate results for large networks although accuracy is difficult to be checked. Decomposition principle, one of most commonly used approach is implemented in three steps:

- Network decomposition into subsystems
- Analysis of each subsystem in isolation
- Analysis of the new aggregated system

The network is broken down to nodes (single servers or multiservers) and a system of equations for the underlying Markov process is developed (Praphamontripong *et al*., 2006). 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. 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 (Lee and Pollock, 1987a, b; Nikolov, 2013). In this study, researchers develop model of asynchronous web server using decomposition and aggregation.

DESCRIPTION OF THE QUEUING MODEL

Researchers make the same assumptions about the input to the asynchronous web server as by Praphamontripong *et al*. (2006). When read requests and write requests arrive a read/write operations are assigned, initiated and executed. The completion of these operations are handled by a common queue regardless of the request type. Finally, completion events are demultiplexed and dispatched to an appropriate completion event handler. Each request type has its own completion event handler.

The model of the queuing network covering the above operations is illustrated in Fig. 1 (Praphamontripong *et al*., 2006). Event handlers for incoming read and write request are represented by multiservers 1 and 2, common completion server is represented by queue 0 and completion event handlers are represented by servers 3 and 4, respectively. Researchers shall refer below to the multiservers as to merging queues and to queue 0 as to a merger.

Outputs of the two parallel multiservers or merging queues are connected to a single server or merger queue. The arrivals to multiserver i are independent and follow exponential distribution with rate λ_i (i = 1, 2). Each

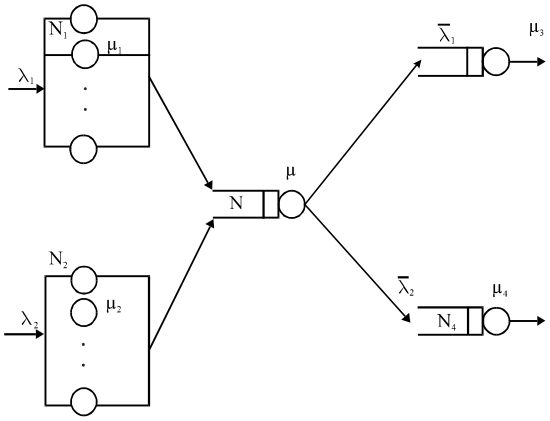

Fig. 1: The model of the queing network

multiserver has N_i single identical units with exponentially distributed service times with rate H_i. Multiservers have no buffers. The service time of the server 0 is also exponentially distributed with rate μ and its buffer size is N. The index i can be viewed as the sequential number of the multiservers and runs from 1-2 unless otherwise stated. The arrivals to any multiserver are dispatched to some of its units if there is one free. When an arrival finds no free server, however, it is lost, or rejected. After completion of the service at multiserver i the request proceeds to server 0 and is taken immediately into service if the server is free (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 the buffer 0 becomes full all incoming arrivals from multiservers i are blocked and the multiservers are forced to wait so that there is no loss of requests at server 0 but some delay is inserted due to the fact 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 decremented by one. As the time progresses the number of units forced to shut down can vary from 0 to N_i, i.e., at some point all units of multiserver i are blocked and it produces no requests to server 0. 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 multiservers by inserting additional time required to wait for joining the merged queue.

If $j = L_1+L_2$ units are blocked by the merger 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. After completion of service at queue 0 requests are splitting and join queue one of the output queues. Presence of the dispatcher in the system secures that that requests from queue 1 (read request) go to server 3 (read request) and requests from queue 2 (write request) go to server 4 (write request).

The approach used is to decompose the system into individual queues and analyze each individual queue separately. Researchers assume that there are Poisson arrivals with rate $\tilde{\lambda}_i^{N_i-L_i}$ from multiserver i to server 0 as long as N_i-L_i units are not blocked. Obviously, arrival rates decrease as the number of the blocked units L_i increases and $\tilde{\lambda}_i^0=0$ that is at some point multiserver 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 $K = N_1+N_2$ places so its total size is N+K. Researchers introduce the following notations: B_m size of the buffer for serverm (m = 3, 4). $N_m = B_m+1$ server capacity (m = 3, 4).

$T_i^{N_i-L_i}$ 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 exponentially distributed random variables each of which has a parameter of μ.

E[T] expected value of the clearance time for queue 0 $f_i^{N_i-L_i}$ P [ith queue is full given N_i-L_i units are operational (not blocked)], $\bar{\lambda}_i^{N_i-L_i}$ arrival rate to the multiserver 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 units blocked, there are N+j units in the extended buffer of server 0, given that units are not blocked. $\bar{\lambda}_i$ is throughput for queue i, $\bar{\lambda}=\bar{\lambda}_1+\bar{\lambda}_2$ is total system throughput $K = N_1+N_2$, $\delta_{m,n}$ is the Kronecker delta symbol:

$$\prod_n^m = 1 \quad \text{if } n>m$$

$$\sum_n^m = 0 \quad \text{if } n>m$$

The system is considered to be balanced and all probabilities in this study are steady-state ones. The states of the queue 0 are represented by the number of the requests in the queue and in service 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. 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:

$$P[k]=P[0]\prod_{j=1}^{k}\frac{\lambda_a(j)}{\mu} \quad (k = 1, ..., N+K) \tag{1}$$

$$\sum_{k=0}^{N+K} P[k]=1 \tag{2}$$

In Eq. 1 and 2, $\lambda_a(j)$ by definition is a transition rate from state j to state j+1 for the aggregated system. The aggregated states of queue 0 follow simple birth and death process (Gokhale *et al*., 2005; Lee and Pollock, 1987a). The relationship between steady-state probabilities of the original queue 0 and those of the aggregated queue 0 is:

$$P[N+n; z_1,..., z_1,..., z_n]=\frac{P[N+n]\prod_{i=1}^{2}\prod_{k=N_i-L_i+1}^{N_i}\tilde{\lambda}_i^{k}}{\Omega_n} \tag{3}$$

Where:

$$\lambda_a=\sum_{i=1}^{2}\tilde{\lambda}_i^{N_i} \quad \text{for } k=1,...,N \tag{4}$$

$$\lambda_a(N+n)=\frac{\Omega_{n+1}}{\Omega_n} \quad \text{for } n = 1, ..., K-1 \tag{5}$$

$$\Omega_n=\sum_{\forall L_n^p}\prod_{i=1}^{2}\binom{n-\sum_{l=1}^{i-1}L_i}{L_i}\prod_{k=N_i-L_i+1}^{N_i}\tilde{\lambda}_i^{k} \tag{6}$$

By $L_n^p = [L_1, L_2]$ researchers denote a vector for which $0\le L_1\le N_1$ and $0\le L_2\le N_2$ and $L_1+L_1 = n$ hold. Superscript p points to a particular vector from the set of such vectors. We also define a 1×n vector:

$$z_n^{p,q} = [z_1, ..., z_1, ..., z_n] \; z_1\in\{1, 2\}$$

for which:

$$\sum_{l=1}^{n}\delta_{i,z_1}=L_i$$

holds for all element of the vector L_n^p. q is a sequential number. In the context of nonaggregated queue 0 each vector $z_n^{p,q}$ represents a unique order of arrivals from multiservers i (i = 1, 2) where the numbers of units from each multiserver are uniquely determined by the vector L_n^p. The occupancy probabilities of the original queue 0 are denoted as P[k] for k = 1, ..., N and:

$$P[N+n; z_n^{p,q}] \quad \text{for } n=1,..., K$$

Applying the conservation flow rule to queue i yields (Praphamontripong *et al*., 2006; Lee and Pollock, 1987a, b):

$$\bar{\lambda}_i^{N_i-L_i}=\tilde{\lambda}_i^{N_i-L_i}\left(\delta_{L_i,0}\sum_{k=0}^{N}P[k]+\sum_{\forall\left(z_n^{p,q}\text{ and }L_n^p(i)=L_i\right)}P\left[N+n; z_1,..., z_1,..., z_n\right]\right) \quad \text{for } L_i = 0, ..., N_i-1 \tag{7}$$

The expected value of the clearance time $E[T_i^{N_i-L_i}]$ (Bhat, 2008) is:

$$E\left[T_i^{N_i-L_i}\right]=\frac{1}{\mu_i}+\sum_{j=0}^{K-1}\alpha_i^{N_i-L_i}[N+j]E[T] \quad \text{for } j=0,...,K \tag{8}$$

Where:

$$\alpha_i^{N_i-L_i}[N+j]=\frac{P[N+j]-b_i^{N_i-L_i}[j]}{1-\sum_{n=1}^{K}b_i^{N_i-L_i}[n]} \tag{9}$$

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

$$\sum_{\forall\left(Z_n^{p,q}\text{ and }L_n^p(n)>L_i\right)}\frac{\prod_{i=1}^{2}\prod_{k=N_i-L_i+1}^{N_i}\tilde{\lambda}_i^k}{\Omega_n}\quad\text{for } L_i=0,\ldots,N_{i-1} \tag{10}$$

$$E[T]=E[T_3]+E[T_4] \tag{11}$$

$$f_i^{N_i-L_i}=P_i^{N_i-Li}[0]\frac{\left(\rho_i^{N_i-L_i}\right)^{N_i-L_i}}{\left(N_i-L_i\right)!} \tag{12}$$

Where:

$$P_i^{N_i-L_i}[0]=\frac{1}{\sum_{k=0}^{N_i-L_i}\frac{\left(\rho_i^{N_i-L_i}\right)^k}{k!}} \tag{13}$$

$$\rho_i^{N_i-L_i}=\lambda E\left[T_i^{N_i-L_i}\right] \tag{14}$$

$$\bar{\lambda}_i^{N_i-L_i}=\lambda_i\left(1-f_i^{N_i-L_i}\right) \tag{15}$$

$$\bar{\lambda}_4=\sum_{L_i=0}^{N_i}\bar{\lambda}_4^{N_i-L_i}\left(\delta_{L_i,0}\sum_{k=0}^{N}P[k]+\sum_{\forall\left(z_n^{p,q}\text{ and }L_n^p(i)=L_i\right)}P[N+n;z_1,\ldots,z_1,\ldots,z_n]\right)\quad\text{for } L_i=0,\ldots,N_{i-1} \tag{16}$$

For the splitting queues (3 and 4) the following equations apply:

$$P[N_m]=\frac{(1-\rho_m)\rho_m^{N_m}}{\left(1-\rho_m^{N_m+2}\right)} \tag{16}$$

$$E[T_m]=\frac{1}{\mu}+\frac{P[N_m]}{(1-P[N_m]\rho_m)\mu_m} \tag{17}$$

$$\bar{\lambda}_m=\bar{\lambda}_m\left(1-P[N_m]\rho_m\right) \tag{18}$$

Where:

$$\rho_m=\frac{\bar{\lambda}_m}{\mu_m}\quad\text{for } m=3,4$$

Since, a conservation flow is enforced $\bar{\lambda}_3=\bar{\lambda}_4$ and $\bar{\lambda}_4=\bar{\lambda}_2$ (Lee and Pollock, 1987b; Nikolov, 2013). Solving simultaneously Eq. 1-18 produces the wanted performance metrics of the configuration.

Table 1: The probabilty of losing request

N	$N_1=N_2=1$	$N_1=N_2=2$	$N_1=N_2=3$	$N_1=N_2=4$	$N_1=N_2=5$
2	0.09692	0.005086	0.001974	0.0004855	0.0001127
3	0.09191	0.004792	0.001623	0.0004762	0.0001121
4	0.09109	0.004571	0.001619	0.0004753	0.0001110
5	0.09094	0.004534	0.001611	0.0004751	0.0001009
6	0.09091	0.004527	0.001609	0.0004750	0.00009083
7	0.09091	0.004525	0.001609	0.0004748	0.00009080
8	0.09091	0.004525	0.001607	0.0004746	0.00009061
9	0.09090	0.004524	0.001607	0.0004746	0.00009061
10	0.09090	0.004522	0.001607	0.0004746	0.00009061
11	0.09090	0.004522	0.001604	0.0004744	0.00009061

Table 2: Aggregated probabilties

N	P[N]	P[N+1]	P[N+2]	P[N+3]	P[N+4]
2	0.012870	0.007031	0.00091	9.00×10^{-3}	8.33×10^{-5}
3	0.006300	0.001261	0.00025	3.24×10^{-4}	2.45×10^{-5}
4	0.001269	0.000251	0.00005	6.43×10^{-5}	4.87×10^{-7}
5	0.000976	0.000102	2.21×10^{-4}	1.82×10^{-5}	6.73×10^{-8}
6	0.000724	0.000042	1.02×10^{-4}	6.69×10^{-6}	4.02×10^{-9}
7	0.000136	0.000017	6.23×10^{-5}	2.18×10^{-6}	5.12×10^{-10}
8	0.000087	0.000009	3.17×10^{-5}	6.27×10^{-7}	4.29×10^{-11}
9	0.000053	0.000005	8.78×10^{-6}	1.98×10^{-7}	3.98×10^{-12}
10	0.000021	0.000003	3.27×10^{-6}	7.21×10^{-8}	3.08×10^{-13}
11	0.000009	0.000001	1.09×10^{-6}	4.07×10^{-8}	9.17×10^{-15}

NUMERICAL RESULTS

Input parameters for the experiment are: $\lambda_1 = \lambda_2 = 1$[1/t. u.] time unit, $\mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu = 10$[1/t.u.], $N_3 = N_4 = 6$. Table 1 tabulates the probability of losing (rejecting) requests $\tilde{\lambda}_i / \lambda_i$ as a function of N_i and N. Performance metrics for the read and write requests are identical due to the symmetry. Researchers can observe that increasing the number of the units of the multiservers leads to an essential in some cases by an order of magnitude, reduction of the number of lost requests. Increased size of the buffer of the merger also leads to improvement of this parameter but its impact is not so significant.

Table 2 lists the aggregated probabilities for $N_1 = N_2 = 2$ as a function of N. Impact of the size of merger's buffer is essential for these probabilities, they are changed by orders of magnitude as N increases. researchers also can observe that P[N+n] is much bigger than P[N+n+1] for n = 0, 1, 2, 3, 4 and for large n, P[N+n] even vanishes.

CONCLUSION

The performance analysis of asynchronous web server presented here is easy to use and produces results in a meaningless computing time. It guarantees deep insight into the performance metrics since all probabilities including those of all permutations of the blocked units can be calculated. It can be used at the early design stage before deployment of the product. Numerical results indicate that some probabilities are too small and have little impact on the performance metrics so that the model can be further simplified.

REFERENCES

Balsamo, S., V.D.N. Persone and P. Inverardi, 2002. A review on queueing network models with finite capacity queues for software architectures performance prediction. Perform. Eval., 974: 1-20.

Bhat, U.N., 2008. Introduction to Queuing Theory: Modelling and Analysis in Applications. Springer, Berlin.

Cao, J., M. Andersson, C. Nyberg and M. Kihl, 2003. Web server performance modeling using an M/G/1/K*PS queue. Proc. 10th Int. Conf. Telecommun., 2: 1501-1506.

Gokhale, S.S., A.S. Gokhale and J. Gray, 2005. Response time analysis of a middleware event demultiplexing pattern for network services. Proceedings of the IEEE Global Telecommunications Conference, Volume 2, December 2, 2005, St. Louis, MO., pp: 704.

Lee, H.S. and S.M. Pollock, 1987a. Approximate analysis for the merge configuration of an open queueing network with blocking. Technical Report No. 87-11, University of Michigan, pp: 1-28.

Lee, H.S. and S.M. Pollock, 1987b. Approximate analysis of open exponential queueing networks with blocking: General configuration. Technical Report No. 48-109, University of Michigan, pp: 1-31.

Nikolov, A.V., 2013. Approximate model for a merge configuration of multiservers with finite capacity intermediate buffers. Int. J. Applied Maths., 26: 391-401.

Praphamontripong, U., S. Gokhale, A. Gokhale and J. Gray, 2006. Performance analysis of an asynchronous web server. Proceedings of the 30th Annual International on Computer Software and Applications Conference, Volume 2, September 17-21, 2006, Chicago, IL., pp: 22-28.