Proceedings of the 2011 Winter Simulation Conference
S. Jain, R.R. Creasey, J. Himmelspach, K.P. White, and M. Fu, eds.

MODELING SERVER USAGE FOR ONLINE TICKET SALES

Christine S.M. Currie

University of Southampton
Highfield, Southampton
SO17 1BJ, UK

Lanting Lu

Peninsula College of Medicine & Dentistry
PenTAG/PenCLAHRC
Salmon Pool Lane, EXETER
EX2 4SG, UK

ABSTRACT

This article describes the use of a discrete event simulation model to estimate the server power required for the online sale of concert tickets to a required service standard. Data are available on the number of purchases made per hour and the percentage of tickets booked online for previous concerts and we describe how these are used to estimate the number of users in the system. We use bootstrapping to allow us to take account of the variability in this estimate when calculating the confidence intervals for the simulation model outputs. A queuing model is also introduced, which is useful to provide a quick calculation of how busy the server is before running the more computationally-intensive simulation model. A numerical example is used to describe the model and the methodology.

1 INTRODUCTION

We describe a project that was carried out to estimate the computing power required to serve a web page selling a popular product with a limited shelf life, where the initial demand for the stock is large, e.g. the sale of concert tickets for a high profile act. The vendor has a minimum service rate based on the time taken to complete the purchase and the proportion of buyers who are unable to complete the buying process due to server error.

There is considerable uncertainty about the number of customers who will try to buy tickets in the first few hours that the sale is open for, which tend to be the peak hours for ticket sales. We are incorporating uncertainty from a number of different data sets and wish to include an accurate measure of this uncertainty in our final estimate of the computing power required. Therefore, we use bootstrapping to give a more complete estimate of the variability in the number of users accessing the website at any one time, taking into account all of the different sources of variability . See Cheng and Currie (2010) for more information about bootstrapping.

We consider both a simulation model and a queuing model to describe how customers are progressing through the purchasing process and to obtain the distribution of the number of requests being sent to the server per unit time. Knowledge of this distribution allows us to determine the necessary server capacity for achieving different service levels for the web-based ticketing system. We begin with a brief literature review of previous work modeling the performance of web systems in Section 2 before describing the input modeling and associate bootstrapping in Section 3. The two models used to describe the purchasing process are then described in Sections 4 and 5 and we give some results of the analysis, before concluding in Section 6.

2 LITERATURE REVIEW

The focus of our study is to determine how much server power is required to meet performance targets on server response time for customers and we do not consider the internal workings of the computer system that is processing the ticket sales, i.e. we model performance at a system level rather than a component level. Menascé and Almeida (2002) gives a detailed description of all aspects of capacity planning for web services, within which the models developed can be categorized as system level performance models or component-level performance models.

The basic structure of models of web server performance at a system level is given in Figure 1. Both Cherkasova and Phaal (2002) and Anderson, et al. (2005) use a model of a similar structure to this to model web server performance. The differences between these two models are that 1) the former makes the assumption of deterministic service times and session-based workload, while an arbitrary service time distribution and request-based workload are assumed in the latter; 2) the former places a limit on the number of users in the system while the latter allows a limited number of jobs at one time; 3) the former uses one arrival rate while in the latter a two-state Markov modulated Poisson process is assumed for the arrival process, i.e. two arrival rates are used: one for low traffic and the other for busy traffic; and 4) the former set the service time and maximum number of users as given, while in the latter, the maximum number of jobs that can be processed by the server at the same time and the average service time were obtained through a maximum likelihood estimation using the likelihood function of the observed average response time as discussed in Cao, et al. (2003).

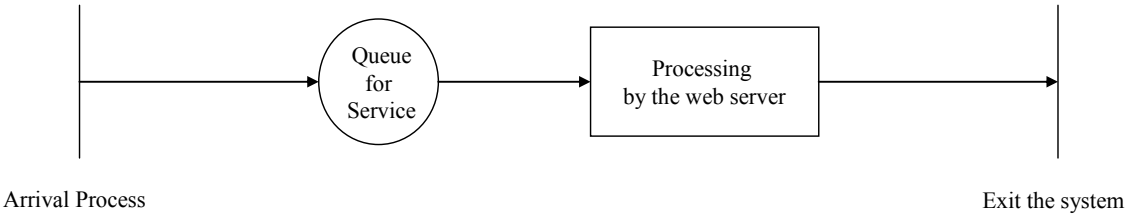

Figure 1: The basic structure of a model of web server performance

When considering a web server system from a component level, more emphasis is placed on modeling the interactions between different components of a web server system. In the majority of studies queuing networks are used, e.g. van der Weij, et al (2009) describe a layered queuing model that is used to find the optimal assignment of threads within the server to minimize server response times; Dilley, et al (1998) use layered queuing models in their evaluation studies of the performance of web servers. Van Der Mei, et al (2001) proposed to model the performance of web servers using a tandem queuing model that consisted of three sub-models. In their study web server performance metrics were predicted using a simulation model, which was validated using measurements obtained in a test lab environment. In Elleithy and Komaralingham (2002), an analytical model of web servers was proposed, but with an assumption that the response time of the queue constantly grows along with the utilization of the server. Wells, et al (2001) analyze the performance of a Colored Petri Nest model of a web server. There are several unknown parameters in their model, which are determined using simulation. Chapters 8 and 9 of Menascé and Almeida (2002) provide an overview of the area of performance modeling of web systems.

3 ESTIMATING RESPONSE IN THE PEAK HOUR

Data are available on the number of ticket purchases made per hour in the first 8 hours of the selling period for a similar concert. The peak hour for ticket sales is usually the first hour and always falls within the first 8 hours and so we do not need to consider data from later in the selling period. We also have data available on the proportion of users who have completed purchases over the phone versus those who have bought their tickets over the Internet for the previous 4 concerts. The real data used in the analysis is confidential; therefore we here use simulated data of a similar form to demonstrate our methodology.

3.1 Responses per Hour

The number of responses per hour for the first 8 hours after tickets go on sale for a previous concert are given in Table 1.

Table 1: Purchases per Hour for First 8 Hours of the Selling Period

Hour	Number of Purchases
1	45893
2	23358
3	8143
4	3258
5	826
6	912
7	519
8	463

We assume that the number of tickets sold in each hour follows a truncated multivariate normal distribution, with the truncation preventing more tickets being sold than the capacity of the concert venue (assumed to be 90,000). We assume that the covariance structure of the multivariate normal distribution matches the covariance structure of a multinomial distribution fitted to these data, i.e. the variance of the number of purchases in hour *i* is equal to

$$Var(X_i) = n(i)\left(1 - \frac{n(i)}{\sum_{k=1}^{8} n(k)}\right),$$

where *n(i)* is the number of tickets sold in hour *i*. The covariance between the numbers of purchases in hours i and j is equal to

$$Cov(X_i, X_j) = \frac{-n(i)n(j)}{\sum_{k=1}^{8} n(k)}.$$

3.2 Proportion of Tickets Sold Online

The proportion of tickets sold online versus on the phone for the previous 4 concerts is given in Table 2.

Table 2: The percentage of tickets sold online for the previous 4 concerts.

Concert	Percentage Sold Online
1	73.7
2	71.4
3	81.8
4	90.2

The data exhibit an upwards trend, which appears to be linear given the few data points available. We can therefore fit a linear regression line to the data to give us a prediction of the online usage for concert 5. We assume normal errors and obtain a result suggesting that 94% (95% prediction interval: 64% - 100%) will purchase tickets online for concert 5. The use of a straight line predictor for this value is a little naïve as at some point in time the linear relationship is likely to break down and the percentage applying online will level off. However, with few data available it seems a reasonable first approximation.

3.3 Combined Probability Distribution for the Online Response During the Peak Hour

We use Bootstrap resampling to combine the uncertainties from the two data sets and obtain an estimate of the number of online purchase requests arriving at the server in the peak hour of the selling period. The bootstrapping allows us to obtain a full probability distribution for the number of online purchases, which will be useful in the modeling stages of the project. We use 1000 iterations of the bootstrapping. In each iteration, we carry out the following procedure:

1. Sample purchase numbers in each of the selling periods from a multivariate normal distribution.
2. Sample the percentage sold online for the previous 4 concerts from univariate normal distributions with mean equal to the prediction from the fitted regression line and variance calculated from the sum of squares of the errors between the data points and the fitted line
3. Find the best fit regression line for the percentage sold online and use this to predict the percentage sold online for the 5^{th} concert, setting any predicted percentages greater than 100% to be equal to 100%
4. Calculate the number of people purchasing tickets online during the peak hour

The bootstrap results can then be used to find a distribution for the number of people purchasing tickets online. For these data we find that the expected number of people purchasing tickets online during the peak hour is 42,300 (95% confidence interval: 32,600 - 46,100).

4 SIMULATION MODELING OF THE ONLINE PURCHASING SYSTEM

The transition diagram for the simulation model of the online purchasing system is given in Figure 2.

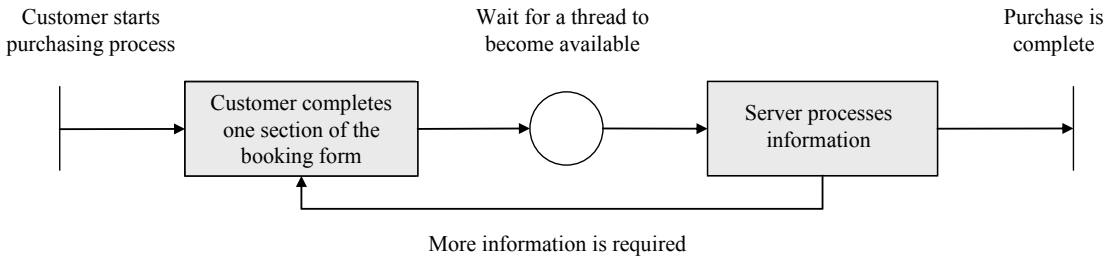

Figure 2: Transition Diagram for the Simulation Model of the Online Purchasing System.

We consider only the peak hour of sales and for each of 25 runs of the simulation, draw the number of customers who wish to purchase a ticket online from the empirical distribution function of the bootstrapping results, obtained as described in the previous section. We use the Trials Calculator in Simul8 (www.simul8.com) to determine the appropriate number of runs such that the variability in the average waiting time is equal to 5% of the mean. This is found to be approximately 25. The Trials Calculator uses AutoSimOA (http://www2.warwick.ac.uk/fac/soc/wbs/projects/autosimoa/), which automates the output analysis. The methodology used by AutoSimOA is described in Hoad, et al (2011).

The aim of the modeling is to determine the number of threads needed in the server to best satisfy the performance requirements that no request should wait longer than 2 seconds and that fewer than 10% of requests should wait longer than 0.5 seconds. More detailed assumptions of the simulation model are given in the following section and the results are given in Table 3.

4.1 Assumptions

- The rate at which one thread responds to one request is independent of the number of threads that are in use (i.e. performance is not dependent on how busy the server is)
- Each piece of information is submitted individually
- The state in which all threads are being used constitutes a state in which some customers are waiting for threads to be available
- The time a user spends working on a request (e.g. completing credit card details) follows an exponential probability distribution with a mean of 30 seconds
- The time the server spends processing each question follows an exponential distribution with a mean of 70 milliseconds (based on expert opinion)
- There are on average 10 pieces of information to be submitted by the user but some users may have to resubmit information because of errors; therefore the probability of exiting the system after server processing is 0.1 and the probability of needing to submit more information is 0.9

4.2 Results

The results given in Table 3 suggest that the optimal number of threads required to meet the required service standard is 10.

Table 3: Results of the Web System Simulation Model (95% Confidence Intervals Given in Parentheses)

Number of Threads	Percentage of Requests Waiting Longer than 0.5s	Maximum Waiting Time for a Request (mins)
8	85.3 [84.6, 86.0]	0.57 [0.55, 0.59]
9	33.0 [26.5,39.4]	0.04 [0.03, 0.05]
10	0.02 [0.00,0.03]	0.01 [0.01, 0.01]

5 SIMULATION MODELING OF A STEADY-STATE WEB SYSTEM

Observations of the simulation model output suggest that the system approaches a steady state approximately 30 minutes after the start of the sales period. For example Figure 3 shows how the number of customers in the process of completing the online form changes since the start of the selling period, demonstrating a leveling off after approximately 25 to 30 minutes. For the steady-state situation, a queuing model can provide us with some simple analytical results much more quickly than the simulation model, which takes some time to run given the large numbers of individuals in the system.

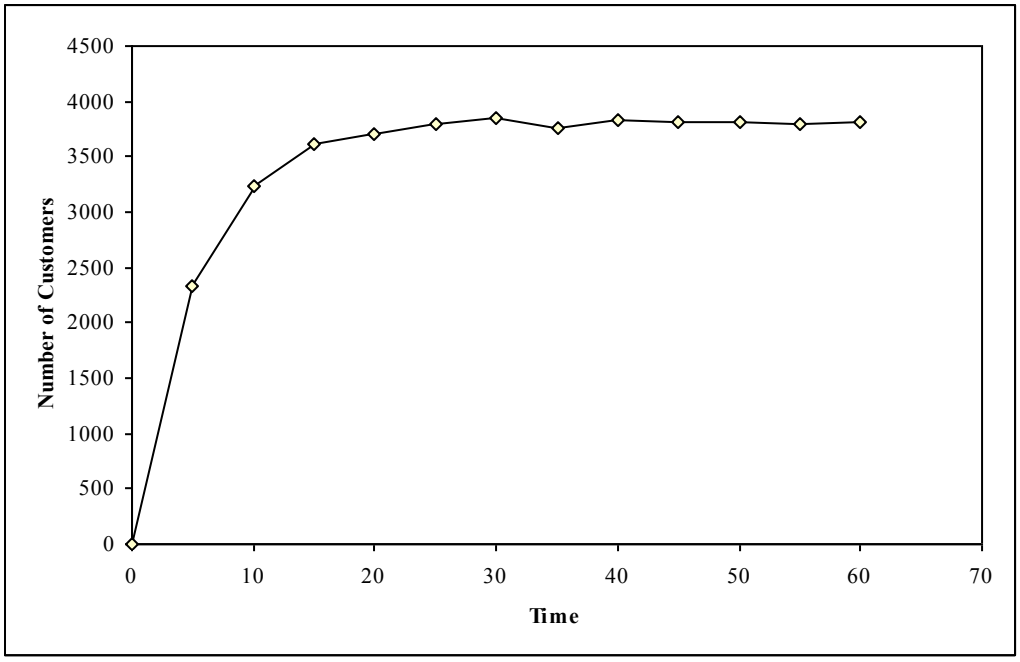

Figure 3: the variation in the number of customers completing the online form over the selling period, where zero time is when the ticketing system is opened.

5.1 The Model

With the assumptions given above in Section 4.1, we can use a queuing model to describe the usage of the web system. We define our system in terms of n, the number pieces of information needing processing by the system. When there are zero threads in use, $n = 0$ and M users will be thinking about submitting information; when there are 10 threads in use, $n = 10$ and $M - 10$ users will be thinking about submitting information. When the system is in a state n that is greater than N, there are n-N users waiting for threads to become available. Figure 4 gives the transition diagram for the system.

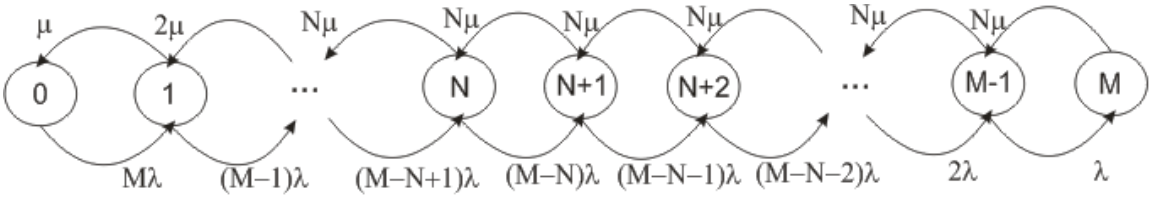

Figure 4: Transition Diagram for the Markov Chain Model of the Web-Based System.

The system can therefore be described as a birth-death process, with $M + 1$ states ($n = 0, 1, 2, \ldots, M$). The rate at which we move from state n to state $n + 1$ is equal to

$$\lambda_n = \begin{cases} (M-n)\lambda & n = 0, 1, \ldots, M-1 \\ 0 & n \geq M \end{cases}$$

and the rate at which we move from state n to state $n - 1$ is equal to

$$\mu_n = \begin{cases} 0 & n = 0 \\ n\mu & n = 1,2,\ldots,N \\ N\mu & n = N+1, N+2,\ldots,M \end{cases}$$

Therefore, the probability that we find the system in state n when we observe it, or alternatively, the proportion of time that the system spends in state n is equal to

$$\pi_n = \begin{cases} \dfrac{M!}{(M-n)!n!}\left(\dfrac{\lambda}{\mu}\right)^n \pi_0 & n = 0,1,2,\ldots,N \\ \dfrac{M!}{(M-n)!N!}\left(\dfrac{\lambda}{\mu}\right)^n \dfrac{1}{N^{n-N}}\pi_0 & n = N+1, N+2,\ldots,M \end{cases},$$

where π_0 is determined by ensuring that the sum of the probabilities of being in each of the states is one. For this problem, π_0 does not have a nice, closed form expression.

5.2 Results

If the steady-state assumption is valid for the system, we can use the queuing model to obtain some simple statistics. When the system is in state $N + 1$ or greater, there will be information waiting to be processed by the server. Therefore, the probability that there are users waiting for the server to process one of their responses will be equal to

$$\Pr(\textit{there are users waiting}) = \sum_{n=N+1}^{M} \pi_n .$$

The expected number of users waiting in a queue can be estimated by

$$\textit{Expected number in queue} = \sum_{n=N+1}^{M} (n-N)\pi_n .$$

Below in Table 4 we give the values of these statistics for three different values of N, the number of threads available. Using the output of the bootstrapping described in Section 3, we are also able to give an indication of the variability in these statistics that reflects the variability in the number of concurrent users, M.

The online form takes approximately 5 minutes to complete; therefore the average number of concurrent users is likely to be closer to 3500 (one twelfth of 42,300) and we divide our estimates of the total users in the first hour by 12 to obtain our estimates of M.

Table 4: Results of the Queue Model (95% Confidence Intervals Given in Parentheses)

Number of Threads	Probability there are Users Waiting	Expected Number in the Queue
8	1.00 [0.342, 1.00]	170 [1.61, 404]
9	0.718 [0.173, 0.927]	9.70 [0.576, 38.5]
10	0.419 [0.0835,0.577]	2.56 [0.226, 5.24]

The results suggest that the system is likely to be too busy with 8 threads to provide a reasonable service but should be able to cope with the demand if there were 10 threads.

6 CONCLUSION

The results presented in the previous section give a suggestion of the number of threads required for the server to meet the suggested performance requirements. The results of the simulation model lead to the same conclusions as those calculated analytically from the queue model, which provides a good check of their accuracy. It also justifies the use of the analytical model to provide an initial, very quick calculation of the state of the system before running the computationally-intensive simulation model.

We have described two models of a web-based purchasing system that allow us to draw out key performance statistics for the sales process. The discrete event simulation model is particularly useful in this application as it allows us to include the initial transient effects on the performance of the web system. Nonetheless, the system appears to enter a steady-state after approximately half an hour and so a steady-state queuing model is justified to give us a quick, rough and ready guide to how busy the server is likely to be before running the more computationally intensive simulation model.

The use of bootstrapping in this application allows us to very easily draw out statistics for our final results that take into account the full variability in our estimate of the number of concurrent users during the peak hour of sales. In this example, our estimate of this key simulation parameter draws on only two different data sets but the methodology would hold in other situations where the process of estimating the simulation input parameters is much more complex.

REFERENCES

Andersson, M., J. Cao, M. Kihl and C. Nyberg. 2005. "Performance modeling of an Apache web server with bursty arrival traffic", *IC'03 : Proceedings of the International Conference on Internet Computing*. ISBN: 1-932415-02-5. Publisher: CSREA Press.

Box, G. E. P., W. Hunter and S. Hunter. 1978. *Statistics for Experimenters: An Introduction to Design, Data Analysis, and Model Building*. Wiley.

Cao, J., M. Andersson, C. Nyberg, and M. Kihl. 2003. "Web Server Performance Modeling Using an M/G/1/K*PS Queue", in *10th International Conference on Telecommunications*, Papeete, Tahiti.

Cheng, R.C.H. and C.S.M. Currie. 2009. "Resampling methods of analysis in simulation studies" In *Proceedings of the 2009 Winter Simulation Conference,* Edited by M.D. Rossetti, R.R. Hill, B. Johansson, A. Dunkin and R.G. Ingalls. Piscataway, New Jersey: Institute of Electrical and Electronics Engineer, Inc.

Cherkasova, L. and P. Phaal. 2002. "Session-based admission control: A mechanism for peak load management of commercial web sites." *IEEE Transactions on computers*, vol. 51, no. 6, pp. 669-685, June 2002.

Dilley, J., R. Friedrich, T. Jin and J. Rolia. 1998. "Web server performance measurement and modeling techniques", *Performance Evaluation*, vol. 33, pp. 5-26, 1998.

Elleithy, K. M. and A. Komaralingam. 2002. "Using a Queuing Model to Analyze the Performance of Web Servers." Available at http://citeseerx.ist.psu.edu/viewdoc/summary?doi=?doi=10.1.1.19.3667.

Hoad, K. A., S. Robinson and R. Davies. 2011. "AutoSimOA: A framework for automated analysis of simulation output." *Journal of Simulation*, vol. 5, pp. 9-24.

Menascé, D. A. and V.A. Almeida. 2002. *Capacity Planning for Web Services*. Prentice Hall, Upper Saddle River, NJ.

Van Der Mei, R. D., R. Hariharan and P.K. Reeser. 2001. "Web server performance modeling." *Telecommunication Systems*, vol. 16, no. 3, 4, pp. 361-378.

Van der Weij, W., S. Bhulai and R. van der Mei. 2009. "Dynamic thread assignment in web server performance optimization", *Performance Evaluation*, vol. 66, pp. 301-310.

Wells, L., S. Christensen, L.M. Kristensen and K.H. Mortensen. 2001. "Simulation based performance analysis of web servers." In *Proceedings of the 9th International Workshop on Petri Nets and Performance Models (PNPM 2001)*. IEEE Computer Society, 2001, pp. 59-68.

AUTHOR BIOGRAPHIES

CHRISTINE CURRIE is a lecturer of Operational Research in the School of Mathematics in the University of Southampton, UK, where she also obtained her Ph.D. She is now Managing Editor for the Journal of Simulation and previously the Book Review editor. Christine has been co-chair of the Simulation Special Interest Group in the UK Operational Research Society for a number of years and involved in the organization of the UK Simulation Workshop. Her research interests include mathematical modeling of epidemics, Bayesian statistics, revenue management, variance reduction methods and optimization of simulation models.

LANTING LU is a research fellow at the Peninsula College of Medicine and Dentistry, affiliated to Exeter University, UK. She has a PhD and MSc in Operational Research from the University of Southampton, UK. Her research interests include simulation modeling of manufacturing processes, health care and classification analyses.