Performance modeling of an Apache web server with bursty arrival traffic

Andersson, Mikael; Cao, Jianhua; Kihl, Maria; Nyberg, Christian

Published in:
IC'03 : proceedings of the international conference on internet computing

2003

Performance Modeling of an Apache Web Server with Bursty Arrival Traffic

Mikael Andersson, Jianhua Cao, Maria Kihl and Christian Nyberg*
Department of Communication Systems, Lund Institute of Technology
Box 118, SE-221 00 Lund, Sweden
Email: {mike, jcao, maria, cn}@telecom.lth.se

Abstract--Performance modeling is an important topic in capacity planning and overload control for web servers. We present a queueing model of an Apache web server that uses bursty arrival traffic. The arrivals of HTTP requests is assumed to be a Markov Modulated Poisson Process and the service discipline of the server is processor sharing. The total number of requests that can be processed at one time is limited to K. We obtain web server performance metrics such as average response time, throughput and blocking probability by simulations. Compared to other models, our model is conceptually simple. The model has been validated through measurements and simulations in our lab. The performance metrics predicted by the model fit well to the experimental outcome.

Keywords--Internet, World Wide Web, web server, performance model, MMPP.

I. Introduction

Performance modeling is an important part of the research area of web servers. Without a correct model of a web server it is difficult to give an accurate prediction of performance metrics. A validated model is the basis of web server capacity planning, where models are used to predict performance in different settings, see Hu et al. [1] or Menascé and Almeida [2]. In this paper we consider the Apache web server, described in e.g. [3] and [5], which is a well-known web server. It is also the most commonly used server according to [5].

Today a web site can receive millions of hits per day and it may become overloaded as the arrival rate exceeds the server capacity. To cope with this, overload control can be used, which means that some requests are allowed to be served by the web server and some are rejected. In this way the web server can achieve reasonable service times for the accepted requests. In overload control investigations for web servers, performance models predict improvements when using a certain overload control strategy, see Widell [6] or Cao and Nyberg [7]. Overload control is a research area of its own, but it is depending on performance models that are valid in the overloaded work region. Several attempts have been made to create performance models for web servers. van der Mei et al. [8] modeled web servers as tandem queueing networks. The model was used to predict web server performance metrics and was validated through measurements and simulations. Wells et al. [9] made a performance analysis of web servers using colored Petri nets. Their model has several parameters, some of which are known. Unknown parameters are determined by simulations. Dilley et al. [10] used layered queueing models in their performance studies. Cherkasova and Phaal [11] used a model similar to the one presented in this paper, but with assumptions of deterministic service times and session-based workload. Beckers et al. [12] proposed a generalized processor sharing performance model for Internet access lines which includes web servers. Their model describes the flow-level characteristics of the traffic carried. They established simple relations between the capacity, the utilization of the access line and download times of Internet objects.

However, several of the previous models are complicated. It lacks a simple model that is still valid in the overloaded work region. A simple model renders a smaller parameter space thus easier to estimate, while a complicated model usually contains parameters that are difficult to obtain.

A simple model like the M/M/1/K or M/D/1/K with a First-Come-First-Served (FCFS) service discipline can predict web server performance quite well. But conceptually it is difficult to assume that the service distribution is exponential or deterministic and that the service discipline is always FCFS.

In this paper we describe a model of the Apache web server [3], that consists of a processor sharing node with a queue attached to it. The total number of jobs in the system is limited. The arrival process to the server is assumed to be a two-state Markov Modulated Poisson Process (MMPP). MMPP's are commonly used to represent bursty arrival traffic to communication systems, such as web servers (Scott et al., [13]). The service time distribution is arbitrary. A system like this is called an MMPP/G/1/K*PS queue. The average service time and the maximum number of jobs are parameters that can be determined through a maximum likelihood estimation, see Cao et al. [14]. By simulating the system, we were able to obtain web server performance metrics such as throughput, average response time and blocking probability.

* This work has been supported by the Swedish Research Council under contract No. 621-2001-3053.

Our validation environment consists of a server and a computer representing clients connected through a switch. The measurements validate the model. Results show that the model can predict the performance metrics at both lighter loaded and overloaded regions.

The rest of the paper is organized as follows: In section II we describe our new web server model. Our model is validated through measurements and simulations in Section III. Section IV shows the results and the discussion. The last section concludes our work.

II. Web Server Model

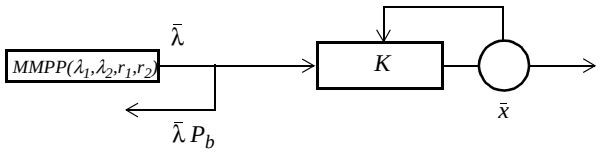

Fig. 1. An MMPP/G/1/K*PS model of a web server

We model the web server using an MMPP/G/1/K*PS queue as Figure 1 shows. The requests arrive according to a two-state Markov Modulated Poisson Process (MMPP) with parameters λ_1, λ_2, r_1, r_2. An MMPP is a doubly stochastic Poisson process where the rate process is determined by a continuous-time Markov chain. A two-state MMPP (also known as MMPP-2) means that the Markov chain consists of two different states, S_1 and S_2. The Markov chain changes state from S_1 to S_2 with intensity r_1, and transits back with intensity r_2. When the MMPP is in state S_1, the arrival process is a Poisson process with rate λ_1, and when the MMPP is in state S_2, rate λ_2 is used, according to Figure 2.

The mean rate $\bar{\lambda}$ and the variance v in a two-state MMPP are given as follows, see e.g. Heffes [15]:

$$\bar{\lambda} = \frac{\lambda_1 r_2 + \lambda_2 r_1}{r_1 + r_2} \quad (1)$$

and

$$v = \frac{r_1 r_2 (\lambda_1 - \lambda_2)^2}{(r_1 + r_2)^2}$$

The service time requirements for jobs in the queue have a general distribution with mean $\bar{x}$. A job in the queue receives a small quantum of service and is then suspended until every other job has received an identical quantum of service in a round-robin fashion. When a job has received the amount of service required, it leaves the queue.

The service can handle at most K requests at a time. A request will be blocked if the number K has been reached. The probability of blocking is denoted as P_b. The rate of blocked requests is given by $\bar{\lambda} P_b$. The throughput H is the rate of completed requests and the average response time T is the expected sojourn time of a job. The average response time, throughput and blocking probability (T, H, P_b) are performance metrics that can be obtained by simulations.

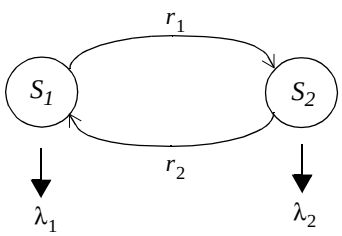

Fig. 2. The MMPP-2 state machine

III. Experiments

A. MMPP parameters

To be able to use the MMPP in our experiments, its parameters had to be determined. We chose to set the mean arrival rate for the MMPP process, and then determine MMPP parameters from that value. r_1 and r_2 were set to 0.05 and 0.95 respectively. The low rate, λ_1, was set to

$$\lambda_1 = 0{,}75 \cdot \bar{\lambda}$$

Equation (1) then gives λ_2:

$$\lambda_2 = \frac{((r_1 + r_2) \cdot \bar{\lambda} - \lambda_1 r_2)}{r_1} \quad (2)$$

This means that λ_2 is a high rate and that it can be seen as a sudden burst rate. λ_2 will be used 5% of the time according to the settings of r_1 and r_2. The parameters have been set this way in order to simulate bursty traffic with random peaks in the arrival rate in both measurements and simulations.

B. Measurements

Our validation measurements used one server computer and a client computer connected through a 100 Mbits/s Ethernet switch. The server was a PC Pentium III 1700 MHz with 512 MB RAM. The computer representing the clients was a PC Pentium II 400 MHz with 256 MB RAM. Both computers used RedHat Linux 7.3 as operating system. Apache 1.3.9 [3] was installed in the server. We used the default configuration of Apache, except for the maximum number of connections. The client computers were installed with a HTTP load generator, which was a modified version of S-Client [16]. The S-Client is able to generate high request rates even with few client computers by aborting TCP connection attempts that take too long time. The original version of S-Client uses deterministic waiting times between requests. We modified the code to use an MMPP arrival process instead, as described above.

The client program was programmed to request dynamically generated HTML files from the server. The CGI script was written in Perl. It generates a fix number N_r of random numbers, adds them together and returns the summation. By varying N_r we could simulate different loads on the web server.

We were interested in the following performance metrics: average response time, throughput and blocking proba-

bility. The throughput was estimated by taking the ratio between the total number of successful replies and the time span of the measurement. The response time is the time difference between when a request is sent and when a successful reply is fully received. The average response time was calculated as the sample mean of the response times after removing transients. An HTTP request sent by a client computer will be blocked either when the maximum number of connections, denoted as $N_{conn,max}$, in the server has been reached or the TCP connection is timed out at the client computer. A TCP connection will be timed out by a client computer when it takes too long time for the server to return an ACK of the TCP SYN-request. The blocking probability was then estimated as the ratio between the number of blocking events and the number of connection attempts in a measurement period.

We carried out the experiments in four cases by varying N_r and $N_{conn,max}$. Table 1 shows the configurations of the four experiments: A1, A2, B1 and B2. The performance metrics were collected while the mean arrival rate (in number of requests/second) was changed from 20 to 300 with step size 20.

TABLE 1
THE CONFIGURATION OF THE FOUR EXPERIMENTS

	N_r = 1000	N_r = 2000
$N_{conn,max}$ = 75	A1	B1
$N_{conn,max}$ = 150	A2	B2

C. Simulations

The results from the measurements were compared with the performance metrics from simulations. The system was implemented in a discrete event simulation program written in Java.

When it comes to $\bar{x}$ and K in the model, we used the same parameters that were found in [14], where a similar model was used, with Poissonian arrivals instead of MMPP arrivals.The parameters can be seen in Table 2. The parameters were obtained by a maximizing the log-likelihood function of the observed average response time in [14].
Note that it is the same parameters, that is, no new parameters have been obtained. If the model is correct, it should be possible to use these parameters.

IV. RESULTS AND DISCUSSION

In all experiments, we notice that the performance metrics predicted by simulations fit well to the measurements. Taking into account that the parameters used in the simulations are estimated from different experiments with the same server configurations but different arrival processes, the small discrepancy between our simulations and measurements is acceptable.

Using the estimated parameters from [14], we predicted the web server's performance with discrete event simulations and compared it with corresponding measurements. Figure 3 shows the average response time, the throughput and the blocking probability curves. We also show 95% confidence intervals for the average response time curves.

TABLE 2
THE PARAMETERS USED IN THE SIMULATIONS

	A1	A2	B1	B2
$\hat{\bar{x}}$	0.00708	0.00708	0.00866	0.00834
$\hat{K}$	208	286	215	298

Since the task of B1 and B2 is more computational intensive than that of A1 and A2, the average response time of A1 and A2 is lower than that of B1 and B2. See Figure 3(a) where the server is limited to the same maximal number of requests. The throughput is affected too. When $N_{conn,max}$ is set to the same, the throughput of A1 and A2 is greater than that of B1 and B2.

The performance of the web server is also subjected to $N_{conn,max}$ when the same task is running on the server. The greater $N_{conn,max}$ is, the higher the response time will be when heavy traffic is offered. Figure 3(a) and (b) show how the $N_{conn,max}$ property works in Apache. At a certain rate, Apache allows no more processes to be started, which results in a limited worst case response time. However, the maximum throughput of the server is not conditioned on $N_{conn,max}$. That is because the throughput of the web server under high request rates is determined by the server speed and the processing requirement of each job, not by $N_{conn,max}$.

V. CONCLUSIONS

We have presented a queueing model of an Apache web server, using a bursty arrival process. We obtained web server performance metrics such as average response time, throughput and blocking probability through simulations. We validated the model through four sets of experiments which included measurements and simulations with bursty arrival traffic. The performance metrics predicted by the model fitted well to the measurements.

Future work will include more validation under different types of loads such as network intensive and hard-disk intensive cases. It would also be interesting to see how well the model fits web servers that use an event-driven approach instead of multi-threading.

REFERENCES

[1] J. Hu, S. Mungee and D. Schmidt, "Principles for developing and measuring high-performance web servers over ATM", in Proceedings of INFOCOM '98, March/April 1998, 1998

[2] D. A. Menascé and V. A. F. Almeida, Capacity Planning for Web Services. Prentice Hall, 2002.

[3] "Apache web server", http://www.apache.org

[4] Y. Hu, A. Nanda, Q. Yang, "Measurement, Analysis and Performance Improvement of the Apace Web Server", in 18th IEEE International Performance Computing and Communications Conference, 1999, Phoenix.

[5] Netcraft Web Server Survey, http://www.netcraft.com/survey/archive.html

[6] N. Widell, "Performance of distributed information systems", Department of Communication Systems, Lund Institute of Technology, Tech. Rep. 144, 2002, lic. Thesis.

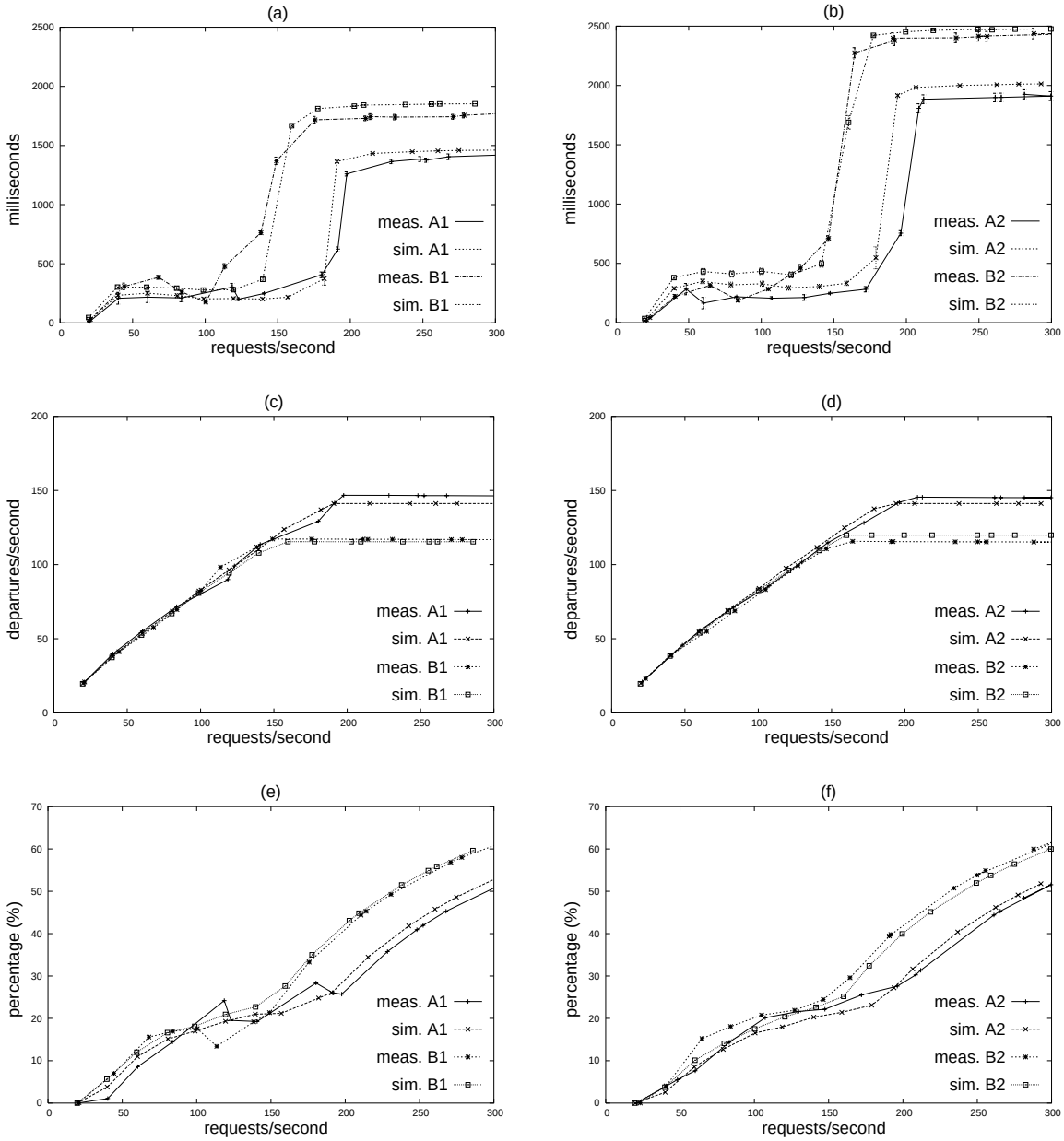

Fig. 3. (a) Average response time of A1 and B1 with 95% confidence intervals. (b) Average response time of A2 and B2 with 95% confidence intervals. (c) Throughput of A1 and B1. (d) Throughput of A2 and B2. (e) Blocking probability of A1 and B1. (f) Blocking probability of A2 and B2.

[7] J. Cao and C. Nyberg, "On overload control through queue length for web servers", in 16th Nordic Teletraffic Seminar, 2002, Esboo, Finland.

[8] R. D. V. D. Mei, R. Hariharan and P. K. Reeser, "Web server performance modeling", Telecommunication Systems, vol. 16, no. 3,4, pp. 361-378, 2001.

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

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

[11] L. Cherkasova and P. Phaal, "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

[12] J. Beckers, I. Hendrawan, R. E. Kooij, and R. van der Mei, "Generalized processor sharing performance model for internet access lines", in 9th IFIP Conference on Performance Modelling and Evaluation of ATM and IP Networks, 2001, Budapest.

[13] S. L. Scott, P. Smyth, "The Markov Modulated Poisson Process and Markov Poisson Cascade with Applications to Web Traffic Modelling", Bayesian Statistics, Oxford University Press, 2003.

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

[15] H. Heffes, "A Class of Data Traffic Processes - Covariance Function Characterization and Related Queuing Results", The Bell System Technical Journal, Vol. 59, No. 6, July-August, 1980.

[16] G. Banga and P. Druschel, "Measuring the capacity of a web server", in USENIX Symposium on Internet Technologies and Systems, December 1997, pp. 61-71.