Performance Modeling of Database Servers in a Telecommunication Service Management System

Maria Kihl[1], Payam Amani[1], Anders Robertsson[2], Gabriela Radu[1], Manfred Dellkrantz[1], and Bertil Aspernäs[3]
[1]Dept. of Electrical and Information Technology, Lund University, Sweden
[2]Dept. of Automatic Control, Lund University, Sweden
[3]Ericsson AB, Karlskrona, Sweden
{maria.kihl, payam.amani, manfred.dellkrantz}@eit.lth.se
luminita.radu@gmail.com, andersro@control.lth.se, bertil.aspernas@ericsson.com

***Abstract*— Resource optimization mechanisms, as admission control and traffic management, require accurate performance models that capture the dynamics of the system during high loads. The main objective of this paper is to develop an accurate performance model for database servers in a telecommunication service management system. We investigate the use of a server model with load dependency. Concurrent requests add load to the system and decrease the server capacity. We derive explicit equations for the state probabilities, the average number of jobs in the system and the average response times. Further, we present some heuristics on how to tune the parameters for given measurement data. Also, using testbed experiments, we validate that the model accurately captures the dynamics of a database server with write-heavy workload.**

***Index Terms*— Performance management; telecommunication systems; queuing theory; database servers.**

I. INTRODUCTION

Resource management of server systems has gained much attention in the last years, since poorly managed resources can severely degrade the performance of a computer system. The experience is that enterprise servers are often the bottlenecks, whereas the network backbone is often underutilized. Therefore, the server systems must provide performance guarantees which satisfy the service-level agreements (on delay, QoS, etc). Also, the system must provide graceful performance degradation during overload.

However, all optimization techniques require accurate performance models of the involved computing systems. The operation region is mainly high traffic load scenarios, which means that the computing systems show non-linear dynamics that needs to be characterized accurately. A software system is basically a network of queues, as examples, the CPU ready queue, semaphore queues, socket queues, and I/O device queues, which store requests in waiting of service in the processors. Therefore, queuing models can be used when describing the dynamic behavior of server systems [1][2][3][4].

In a previous work [1], we have shown that web servers with dynamic content can be modeled as single server queuing systems with processor sharing, where the high load dynamics can be captured with an M/M/1 system. However, this result is only valid for systems with CPU intensive workload. Some recent experiments on databases have shown that the high load dynamics of database servers are completely different for queries involving write operations [5]. Since database servers are important components in Telecommunication service management systems, it is, therefore, important to develop new models for database servers with write-heavy operations.

The concept of load dependent server (LDS) models in which the response time of the jobs in the system is a function of the service time of the jobs and current number of jobs waiting to be serviced has, to the best of our knowledge, firstly been introduced in [6]. Rak *et al.* [7], Curiel *et al.* [8] and Perros *et al.* [6] used standard benchmarks for workload generation and also regression models to capture the system dynamics. A multi-step model parameter calibration strategy was used for fine tuning of the parameters in the model. The resulting models were classified as data driven models. Mathur and Apte [9] presented a queuing network model which represents the load dependent behavior of the LDS. Their model was not analyzed theoretically and was only validated with simulations. A theoretical analysis of the D/G/1 and M/G/1 models with load dependency assumptions was presented in [10] by Leung. These models were developed to be used in congestion control in broadband networks.

In this paper, we add the load dependency behavior to an M/M/m model. The steady state probabilities, average number of jobs in the system and average response times are determined using queuing theory. Also, we perform the same analysis for the case where the queue is limited. The model has a simple structure and can be tuned for various load and database configurations.

Further, in order to tune the parameters of the model to represent the current database and load setup, effects of variations of each parameter on the mean response time of the queries sent to the database as a function of mean effective

 ISBN: 978-1-61208-193-9

arrival rate has been studied. Furthermore, some heuristics for tuning the model parameters are introduced. By means of these heuristics, the model parameters can be tuned to match the measurements from the database in a few steps. Finally,

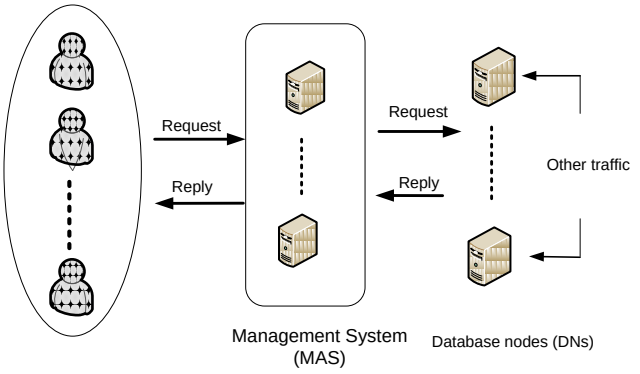

Figure 1. Telecommunication service management systems.

experimental results show that this model is able to capture the high load dynamics of the database server.

This paper is organized as follows. The description of the system is introduced in Section II. Section III is dedicated to introduction of the load-dependent server model. Experiment setups and results are shown in Section IV. Section V concludes the paper.

II. SYSTEM AND PROBLEM DESCRIPTION

In telecommunication service oriented architectures, as mobile networks, all services, either user services as telephony, or administrative services as location updates or billing, are handled by a service management system with its own networks and protocols, as illustrated in Figure 1. The service management systems have a complex architecture, usually implemented as large distributed server systems, with Management application servers (MAS) processing service requests from the telecom networks, and databases (DNs) storing subscriber and service data. The DNs are loaded with service traffic from other networks. All signaling is performed across IP networks, with standardized application layer protocols, as Lightweight Directory Access Protocol (LDAP) or Simple Object Access Protocol (SOAP). The system is required to have high reliability for varying traffic loads, where the DNs may be overloaded by the traffic coming from other networks. The nodes can be owned by different network operators, limiting the available information of traffic loads and service progress. All signaling is performed across IP networks.

In this paper, we focus on the modeling aspects of database servers in telecommunication service management systems. The objective is to develop a performance model for the database server that captures the dynamics during high loads. The performance model can be used in resource optimization schemes, as admission control systems, in order to maximize the throughput of the database server, while keeping some latency constraints. One of the challenges for these database servers is that they have a write-heavy workload, which means that the CPU is not the bottleneck during high loads. This means that previous work on performance modeling of server systems is not applicable since they assume CPU-intensive workload.

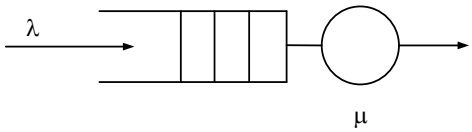

Figure 2. M/M/1 model.

III. LOAD-DEPENDENT SERVER MODEL

Performance models are aimed to be used in the process of designing management entities for server systems. Therefore, the performance model should capture the dominant dynamics of the server system. Most service performance metrics such as response time, service rate and processing delay depend on queue state dynamics.

A. Single server queues

For the objective of performance control, simple models, based on the assumption of a single server queue, are often preferred. The model should only capture the dominating load dynamics of the system, since a well-designed control system can handle many model uncertainties [11]. The classical M/M/1 model, where a single-server queue processes requests that arrive according to a Poisson process with exponential distributed service times, see Figure 2, has been shown to accurately capture the response time dynamics of a web server system [1].

B. M/M/m model with load dependency (M/M/m-LDS)

In this paper, we propose to add load dependency to an M/M/m system. In all load-dependent server models, the service time for a request will be dependent on the number of concurrent requests in the system. This load dependency will model effects of the operating system, memory use, etc., which may cause service degradation when there are many concurrent jobs in a computing system [8]. In the experiment section, we will show that the M/M/m-LDS model accurately captures the behavior of write-heavy workload.

The properties of the load dependent M/M/m model (*M/M/m-LDS*) are set by an exponential distributed base processing time $x_{base} = 1/\mu$ and a dependency factor (f). When a request enters the system, it gets the base processing time x_{base} assigned to it. A single request in the system will always have a processing time of x_{base}. Each additional request inside the system increases the residual work for all requests inside the system (including itself) by a percentage equal to the dependency factor f. When a request leaves the system all other requests have their residual work decreased by f percent again. This means that if n concurrent requests enter the system at the same point, they will all have a processing time of

$$x_s(n) = x_{base} \cdot (1+f)^{n-1} \qquad (2)$$

A special case is when $f = 0$. It means that there is no load dependency, and all requests will have processing time x_{base}.

The system can process a maximum of m concurrent requests at each time instance. Any additional request will

 ISBN: 978-1-61208-193-9

have to wait in the queue. New requests arrive according to a Poisson process with average rate λ.

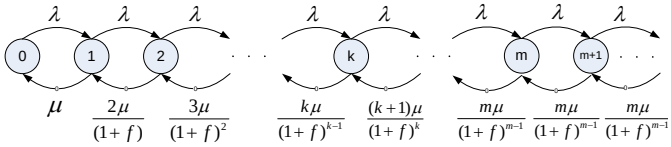

Figure 3. Illustration of M/M/m-LDS model as a Markov chain.

Therefore, the system can be modeled as a Markov chain as illustrated in Figure 3.

The average service rate of the system that depends on the number of concurrent requests in the system, is derived as followed:

$$\mu_k = \begin{cases} \dfrac{k\mu}{(1+f)^{k-1}} & if \quad 0 < k < m \\ \dfrac{m\mu}{(1+f)^{m-1}} & if\ k \geq m \end{cases} \tag{3}$$

By solving the balance equations, stationary probability distribution of existence of *k* concurrent requests in the system is calculated as below:

$$\pi_k = \begin{cases} \dfrac{\left(\frac{\lambda}{\mu}\right)^k}{k!}(1+f)^{\frac{k(k-1)}{2}}\pi_0 & if \quad 0 < k < m \\ \dfrac{\left(\frac{\lambda}{\mu}\right)^k}{m^{k-m}\cdot m!}(1+f)^{(m-1)(k-\frac{m}{2})}\pi_0 & if\ k \geq m \end{cases} \tag{4}$$

As the sum of the probabilities of all possible states equals to one, π_0 can be derived as follows:

$$\sum_{k=0}^{\infty}\pi_k = 1 \rightarrow$$
$$\pi_0 = \frac{1}{1+\sum_{k=1}^{m-1}\frac{\left(\frac{\lambda}{\mu}\right)^k}{k!}(1+f)^{\frac{k(k-1)}{2}} + \frac{\mu\left(\frac{\lambda}{\mu}\right)^m(1+f)^{\frac{m(m-1)}{2}}}{(m-1)!(\mu m-\lambda(1+f)^{m-1})}} \tag{5}$$

The stability condition in this case is

$$\frac{\lambda}{\mu m}(1+f)^{m-1} < 1 \tag{6}$$

The average number of requests in the system, *N*, can be calculated as below:

$$N = \sum_{k=1}^{\infty} k\cdot\pi_k = N_1 + N_2$$
$$N_1 = \sum_{k=0}^{m-1}\frac{\left(\frac{\lambda}{\mu}\right)^k(1+f)^{\frac{k(k-1)}{2}}}{(k-1)!}\pi_0 \tag{7}$$
$$N_2 = \frac{\left(\frac{\lambda}{\mu}\right)^m(1+f)^{\frac{m(m-1)}{2}}(\mu m^2-\lambda(m-1)(1+f)^{m-1})\mu}{(m-1)!(m\mu-\lambda(1+f)^{m-1})^2}\pi_0$$

Finally by means of Little's theorem, the average time each request spends in the system, *T*, can be derived as follows.

$$T = \frac{N}{\lambda} \tag{8}$$

C. M/M/m/n model with load dependency (M/M/m/n-LDS)

In case that the queue is limited to *n* positions, the probability for an empty system, π_0, can be determined as follows. This queuing system is named as *M/M/m/n-LDS*.

$$\pi_0 = \frac{1}{I + II + III}$$
$$I = 1+\sum_{k=1}^{m-1}\frac{\left(\frac{\lambda}{\mu}\right)^k(1+f)^{\frac{1}{2}k(k-1)}}{k!} \tag{9}$$
$$II = \frac{(1+f)^{\frac{1}{2}m^2+\frac{1}{2}m+mn-n-1}\lambda^{n+m+1}}{m^n\mu^{n+m}m!(\lambda(1+f)^{m-1}-\mu m)}$$
$$III = -\frac{(1+f)^{\frac{1}{2}m(m-1)}\lambda^m}{\mu^{m-1}(m-1)!(\lambda(1+f)^{m-1}-\mu m)}$$

Further, the average number of requests in the system is as follows:

$$N = N_1 - N_2$$
$$N_1 = \sum_{k=0}^{m-1}\frac{k\left(\frac{\lambda}{\mu}\right)^k(1+f)^{\frac{1}{2}k(k-1)}\cdot\pi_0}{k!}$$
$$N_2 = \frac{\mu(1+f)^{\frac{1}{2}m^2-\frac{1}{2}m-1}}{m^{m-1}\left(-\lambda(1+f)^{m-1}+\mu m\right)}\cdot\frac{N_{2_{n1}}+N_{2_{n2}}-N_{2_{n3}}}{N_{2_{D1}}+N_{2_{D2}}+N_{2_{D3}}}$$
$$N_{2_{n1}} = -\lambda(n+m)(1+f)^{\left(\frac{1}{2}m^2+\frac{3}{2}m+mn-n-1\right)}\left(\frac{\lambda}{\mu}\right)^{n+m+1}\left(\frac{1}{m}\right)^{n+1}$$
$$N_{2_{n2}} = \left(m(1+f)^m\mu(n+m+1)(1+f)^{\left(\frac{1}{2}m^2+\frac{1}{2}m+mn-n\right)}\right)\left(\frac{\lambda}{\mu}\right)^{n+m+1}\left(\frac{1}{m}\right)^{n+1}$$
$$N_{2_{n3}} = \left(-\lambda(1+f)^m\mu(m-1)+(1+f)\mu m^2\right)\left(\frac{\lambda}{\mu}\right)^m(1+f)^{\frac{1}{2}m(m-1)}$$
$$N_{2_{D1}} = \left(\frac{1}{m}\right)^m(1+f)^{\frac{1}{2}m(m-1)}m!\left(-\lambda(1+f)^{m-1}+\mu m\right)\left(\sum_{k=1}^{m-1}\frac{\left(\frac{\lambda}{\mu}\right)^k(1+f)^{\frac{1}{2}k(k-1)}}{k!}\right) \tag{10}$$
$$N_{2_{D2}} = \left(\frac{-\lambda(1+f)^{m^2+mn-n-1}}{(\mu m)^{n+m}}\right)+\left(\frac{1}{m}\right)^m(1+f)^{\frac{1}{2}m(m-1)}m!\left(-\lambda(1+f)^{m-1}+\mu m\right)$$
$$N_{2_{D3}} = \mu m\left(\frac{\lambda(1+f)^{m-1}}{\mu m}\right)^m$$

Finally, the average response time for a request can be derived using Little's theorem.

D. Parameter tuning

In a telecom system with latency constraints, the dominant dynamic of the system is often characterized by the average response time, *T*, when varying the average arrival rate, *λ*. Tuning of the parameters of the load dependent server model

in a way that it fits the measured data from the actual server system is a necessary step in modeling of such systems. Assuming that λ and *T* are measureable, and therefore, known, there are three main parameters for the M/M/m-LDS model, *m*, *f* and μ, to tune in order to fit the model on the measured data. Further, for the M/M/m/n-LDS there is an extra parameter, *n*, to tune. Therefore, in Figures 4-8, we have illustrated the effect of changing model parameters. In the rest of the paper, this graph will be called the *λ/T* graph. In each figure, we have assumed that two (three) of the parameters are fixed and the one that is mentioned is the variable. As the equations for calculating the mean response time, is rather complex and the parameters are interdependent, more than one set of parameters can be fit on the measured data. Thus using these figures, we can achieve a heuristic rule for tuning the parameters of the load dependent server model.

In the cases where the M/M/m-LDS model is used, the first parameter to be tuned is the number of servers, *m*. As it can be seen in Figure 4, by increasing the maximum number of concurrent requests that can be processed in the system, the linear part of the λ/T graph will be shorter and the exponential rising rate of the graph is increased. In this case it is assumed that (*f*, μ) = (0.7, 22).

The second parameter to be tuned is the dependency factor, *f*. As shown in Figure 5, by decreasing the dependency factor, the linear part of the λ/T graph is increased, however, the change is slower than in the case where *m* is decreased. On the other hand the exponential rising rate of the graph is increased in comparison with the case where *m* is decreased. Here, it is assumed that (*m*, μ) = (3, 22).

The effects of changing μ on the λ/T graph while fixing the two other parameters is illustrated in Figure 6. As shown in the figure, by increasing μ in equal steps, the λ/T graph will be shifted to the right in equal steps. In this case, the rate of rising of the graph is decreased. In this case, (*m*, *f*) = (3, 0.7).

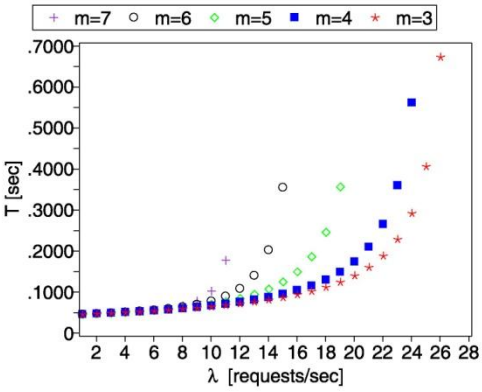

Figure 4. Variations of the λ/T graph for a special scenario with *m* as variable when (*f*,μ) = (0.7, 22).

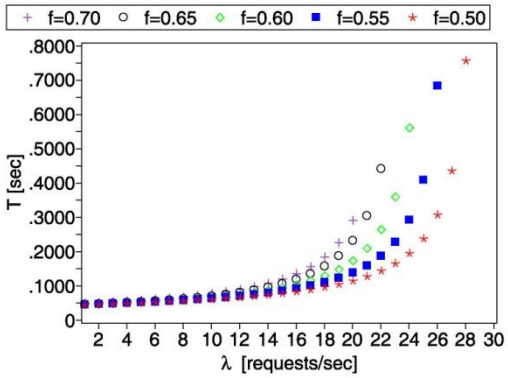

Figure 5. Variations of λ/T graph for a special scenario with *f* as variable when (*m*,μ) = (3, 22).

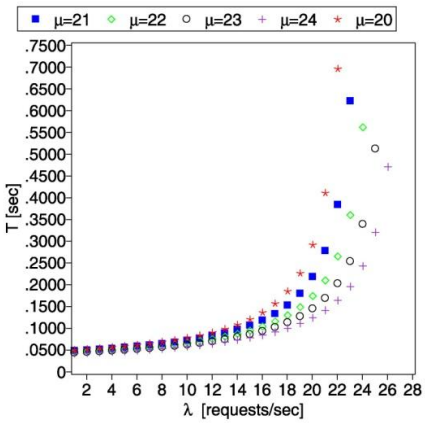

Figure 6. Variations of λ/T graph for a special scenario with μ as variable when (*m*, *f*) = (3, 0.7).

In cases where the M/M/m/n-LDS model is used, there will be a saturation of the response times when the load is high enough to overload the queue. Here it is assumed that the default values are (*m*, *n*, *f*, μ) = (4, 15, 0.6, 22). Figure 7 and Figure 8 show the effects when varying *m* and *f* respectively. In each case the values of the other three parameters are constant. The general effect of changing the parameters is similar as for the case with the infinite queue, with the difference that the response times saturate when the load is high.

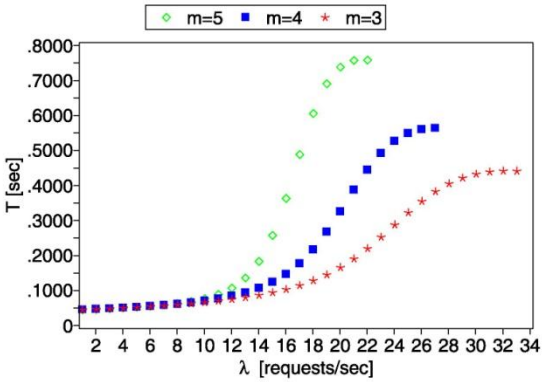

Figure 7. Variations of λ/T graph for a special scenario with *m* as variable when (*n*, *f*, μ) = (15, 0.6, 22).

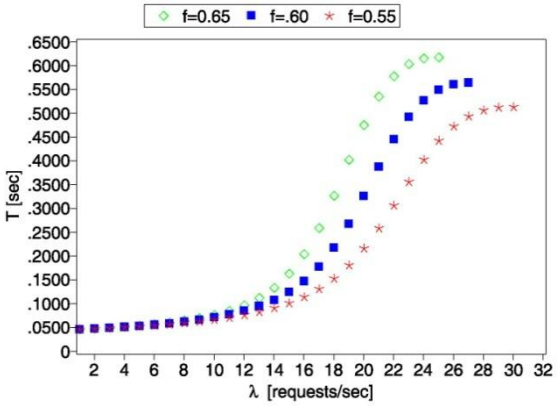

Figure 8. Variations of λ/T graph for a special scenario with *f* as variable when (*m*, *n*, μ) = (4, 15, 22).

IV. EXPERIMENTS AND RESULTS

To validate the proposed model, we have performed a series of experiments in our server lab. We developed a database server testbed with a traffic generator and a MySQL 5.1.41 database server, see Figure 9. The computers were connected to a local Fast Ethernet 100 Mbit/s network.

A. Testbed

The traffic generator was implemented in Java, using the JDBC MySQL connector, and it was executed on a computer with an AMD Phenom II X6 1055T Processor at 2.8 GHz and 4 GB main memory. The operating system is Ubuntu 10.04.2 LTS. The traffic generator used 200 working threads and generates MySQL queries according to a Poisson process with average rate λ queries per second. The behavior of the traffic generator was validated in order to guarantee that it was not a bottleneck in the experiments.

The database server has several relations with the same structure but with different number of tuples. The maximum number of allowed concurrent connections is set to 100. The structure of the relations comes from the Scalable Wisconsin Benchmark [12] with 10 million tuples. Two basic types of queries are used, `SELECT` (read) and `UPDATE` (write).

The queries look like this

```
SELECT * FROM <relation> WHERE unique1=?;
UPDATE <relation> SET unique2=? WHERE unique1=?;
```

The question marks are replaced with uniformly distributed random numbers from zero to ten million.

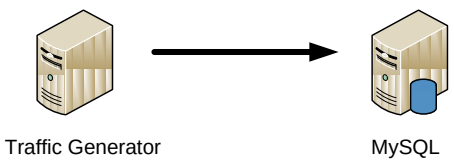

Figure 9. Database server testbed.

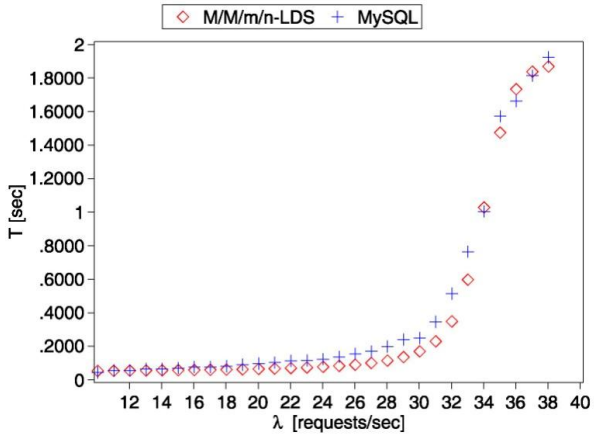

Figure 10. Performance of the M/M/m/n-LDS queuing model in modeling steady state dynamics of a MySQL database server using only update queries.

B. Results

The dynamics of a database server highly depends on the mix of requests, since Select and Update queries require different amount of server capacity. Therefore, we have performed experiments with varying workload mix. Figures 10 and 11 show the results from experiments where the arrival rate is varied from low load to high load. The graphs show the average response times of update queries as a function of the arrival rate. We have fitted M/M/m/n-LDS models for the data using the tuning steps in Section III. In both scenarios, the CPU utilization was very low, also for high loads. The maximum CPU load was about 5%.

In Figure 10, the workload is based on 100% Update queries. The fitted model in this case has the following parameters (*m*, *n*, *f*, μ) = (3, 81, 0.75, 37.1). In order to model the network delays, we have added a bias of 0.023 seconds in the average response times of the proposed model.

Figure 11 depicts the same experiment setup when using a mix of 25% Select and 75% Update queries. The fitted M/M/m/n-LDS model in this case has the following parameters (*m*, *n*, *f*, μ) = (6, 73, 0.44, 35.2). In order to model the network delays, we have added a bias of 0.023 seconds in the average response times of the proposed model.

Figures 10 and 11 verify that the proposed model can represent the average dynamics of a database server with write-heavy workload very well.

V. CONCLUSIONS AND FUTURE WORKS

Resource management schemes require accurate performance models that capture the dominant dynamics of the system in high loads. For server systems with write-heavy workload, load dependent server (LDS) models can be used to model the dynamic overhead effects of concurrent requests. In this paper, queuing theoretic metrics like average number of the requests in the system, average time in the system for each request and the steady state probabilities for M/M/m-LDS models with both unlimited and limited queues have been derived. Further, it has been shown via experiments that the M/M/m/n-LDS model represents the average dynamics of the database server very well. The results are aimed at single

 ISBN: 978-1-61208-193-9

database servers, and not aimed at data centers, which have different dynamics. Furthermore, we will use this model in order to design controllers and state estimators for resource management and admission control of database servers.

VI. ACKNOWLEDGMENTS

The authors at Lund University are members of the Lund Center for Control of Complex Engineering Systems. The work is partly funded by the Swedish Research Council, grant VR 2010-5864.

REFERENCES

[1] J. Cao, M. Andersson, C. Nyberg and M. Kihl, "Web Server Performance Modeling using an M/G/1/K*PS Queue", Proc. of the International Conference on Telecommunication, 2003.

[2] J. Dilley, R. Friedrich, T. Jin, and J. Rolia, "Web server performance measurement and modeling techniques", *Performance Evaluation*, Vol. 33, No. 1, 1998.

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

[4] R. D. van der Mei, R. Hariharan, and P. K. Reeser, "Web server performance modeling", *Telecommunication Systems*, Vol. 16, No. 3, 2001.

[5] M. Kihl, G. Cedersjö, A. Robertsson, B. Aspernäs, "Performance measurements and modeling of database servers", Sixth International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks, 2011.

[6] H. Perros, Y. Dallery, and G. Pujolle, "Analysis of a queuing network model with class dependent window flow control," INFOCOM '92. Eleventh Annual Joint Conference of the IEEE Computer and Communications Societies, IEEE, pp. 968–977 vol.2, May 1992.

[7] A. Rak, A. Sgueglia, "Instantaneous Load Dependent Servers (iLDS) Model for Web Services," In Proceedings of the International Conference on Complex, Intelligent and Software Intensive Systems, 2010.

[8] Curiel, M. and Puigjaner, R., "Using load dependent servers to reduce the complexity of large client-server simulation models", *Performance Engineering*, Springer-Verlag Berlin Heidelberg, pp. 131-147, 2001.

[9] V. Mathur and V. Apte, "A computational complexity aware model for performance analysis of software servers," in Proc. of Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), IEEE Computer Society, pp. 537–544, 2004.

[10] Kin K. Leung, "Load-dependent service queues with application to congestion control in broadband networks", *Performance Evaluation*, Vol. 50, Issue 1, pp. 27-40, October 2002.

[11] K J. Åström and B. Wittenmark, "*Computer–Controlled Systems*", Upper Saddle River, NJ: Prentice Hall, 1997.

[12] D.J. DeWitt, "The Wisconsin benchmark: Past, present, and future", The Benchmark Handbook for Database and Transaction Processing Systems, 1, 1991.

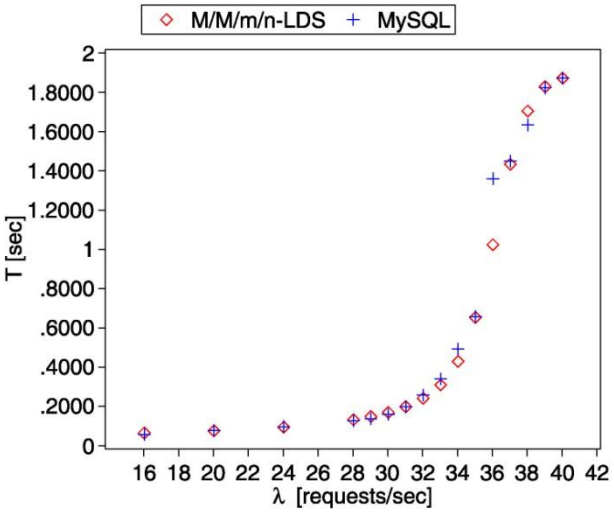

Figure 11. Performance of the M/M/m/n-LDS queuing model in modeling steady state dynamics of a MySQL database server using mixed queries.

 ISBN: 978-1-61208-193-9