An M/G/1 Queue Model for Multiple Applications on Storage Area Networks

Emmanuel Arzuaga and David R. Kaeli
Northeastern University Computer Architecture Research
Electrical and Computer Engineering Department
Northeastern University
360 Huntington Avenue, Boston, MA 02115
{earzuaga,kaeli}@ece.neu.edu

Abstract

Storage Area Networks (SAN) have become one of the most widely used solution for high performance enterprise storage applications. However, the infrastructures of SANs are very complex and varied among different vendors. It is important to understand the behavior of these SANs in order of enhancing performance of the applications that run on them. The main idea of this paper is to understand the behavior of a server-SAN system that manages multiple applications stored in the same volume. We aim at modeling a typical workload of a small company which consists of one license of a database system software and shares that same license for multiple databases. These multiple databases may be used potentially for diverse applications. The objective is then to see how the system would behave when managing queries from different types of databases at the same time. In this paper we will use an $M/G/1$ Queueing Model to analyze the system and will report our first set of experiments which are composed of two different database workloads based on the TPC-C and TPC-H benchmarks. We show that the model can be used to anticipate the impact of growth in the arrival rate on the behavior of the system. This model has the potential of predicting the system behavior when adding more workloads to the mix.

1. Introduction

SANs typically consist of large disk arrays controlled by local storage processors (SPs) and made available to the server as different virtual partitions or logical unit numbers (LUN) through a fast attached media such as fibre channel. In these experiments we want to address the lack of characterization of concurrent workloads running on a server system. A considerable amount research has focused traditionally in characterizing the behavior of streaming data (video/audio) or servers dedicated completely for a single type of application [1]. In real life this may not necessarily be the common case. Due to high cost of database software licensing, small companies often use same licenses to create multiple databases that they will need. For example, companies may have a Web Server that handles online transaction processing (OLTP) type transactions and also internally may need to process decision support systems (DSS) type queries. One possible implementation of their system could be thought as a set of workloads consisting of multiple applications interacting with a single database system. In order of enhancing performance for this type of services, there is the need to fully understand the characteristics of these multiple instance workloads running both single and concurrently.

Analytical Modeling aims at describing a collection of measured and calculated behavior of different computer system components such as workloads, hardware, software, cpu and the storage medium. The measurements are done in a finite period of time, thus we would like this time window to capture the behavior we are interested in. Analytical models can be built for different purposes. Some models are created as a basis for prediction of behavior of certain elements within a system. The impact of moving to faster or slower hardware, for example, can be measured by changing different parameters of each component into the model. Other models, as in our case, are mainly built to gain an understanding of the current activity on the system of interest. We can then measure performance and analyze the behavior of the workloads and the hardware within it.

In this paper we will use an $M/G/1$ Queueing Model to characterize our multiple application system. We can take the advantage that Queueing Theory can be applied to evaluate the efficiency of a computer system that handles requests and provides services. Figure 1 depicts such scheme. The implemented workloads are based on TPC-C (OLTP) and TPC-H (DSS) benchmarks. These benchmarks are specified by the Transaction Processing Council [2]. We used our model with combination of these two types

of workloads as well as each running separately. The rest of

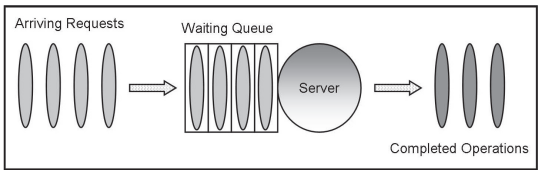


Figure 1. Example of a Queueing System

the paper is organized as follows. In section 2 we start with a description of the $M/G/1$ model. Section 3 will then discuss the characteristics of the implemented workloads. Results of the model simulation and a comparison with real data will be covered in section 4. Section 5 will analyze the results . Section 6 will cover some related work in the area. Then section 7 presents the conclusions and will propose future work.

2. $M/G/1$ Queueing Model

There has been a considerable amount of work analyzing the $M/G/1$ queue system [3, 4, 5, 6].The $M/G/1$ queue has customers or transactions arriving according to a Poisson process with rate λ, but with service times having a general distribution. This provides us with a more general case when comparing it to the $M/M/1$ queue which has Poisson arrival rates and exponential service times. There is a drawback though: the $M/G/1$ queue does not have a general, closed form distribution for the number of transactions in the queue in steady state. However, it provides a generalization based on average values. That is, it gives a general solution for the average number of transactions in the queue, and the application of Little's Theorem provides us the corresponding result for the average time spent in the queue. Collectively, these results are known as the Pollaczek-Khinchin (P-K) formula:

$$T_W = \frac{\rho(1+C_S^2)}{2(1-\rho)}T_S \tag{1}$$

Where $\rho = \lambda T_S$ is the utilization of the system, defined in terms of arrival rate λ and T_S, the average service time. T_W is the average waiting time and C_S^2 is the coefficient of variation squared of T_S. The P-K formula for the $M/G/1$ queue assumes first come first served (FCFS) policy. Using the P-K formula, the average number of transactions in the system Q_L is:

$$Q_L = \rho + \rho^2\frac{(1+C_S^2)}{2(1-\rho)} \tag{2}$$

Table 1. OLTP vs DSS Comparison

Characteristic	OLTP	DSS
Description	Transaction Processing	Informational Processing
Orientation	Transactions	Analysis
Function	Day to day operations	Decision Support, Long-Term informational requirements
Data	Up to date	Consistency maintained over time
Accesses	Read/Write	Read (mostly)
Operations	Index/hash on primary key	Large amount of scans
Performance metric	Transaction Throughput	Query Throughput, response time

and the average waiting line size Q_W is:

$$Q_W = Q_L - \rho \tag{3}$$

$$= \rho^2\frac{(1+C_S^2)}{2(1-\rho)} \tag{4}$$

With these equations, we can build a model that captures the behavior of a system composed of a server attached to a SAN which manages multiple applications in the same volume. We can use input parameters λ, T_S and C_S^2 to estimate system average queue length, utilization and average waiting time. We can then characterize the performance of the system based on these model output.

3. Workload Characteristics

We are using in our experiments two types of workloads; an OLTP and DSS system. These workloads were implemented based on TPC-C and TPC-H [2]. Table 1 shows a comparison of both types of workloads. TPC-C models a wholesale supplier managing orders. Order-entry provides a conceptual model for the benchmark with the underlying components being typical of any OLTP system. The workload consists of five transaction types. Table 2 shows the transaction types and their read/write characteristics. The transactions operate against a relational database composed of 9 tables. These are: Warehouse, District, Customer, History, Order, New-order, Order-Line, Stock, and Item. Transactions will make reads, writes, and rollbacks. The application uses primary and secondary key access. Our OLTP workload consists of warehouses each containing 10 terminals as the TPC-C specification states. The terminal interface consists of a Linux terminal that displays information about the current transaction being performed. The delivery transaction is executed as any other transaction so there is no deferred mode. The TPC-H workload is a complex

Table 2. OLTP Transactions

Name	Access Pattern
New-order	read/write
Payment	read/write
Delivery	read/write
Order-status	read
Stock-level	read

Decision Support workload. TPC-H Benchmark models ad hoc queries. The benchmark consists of queries that perform read accesses to the database and a pair of database update functions (add, delete). The database size is determined from fixed Scale Factors (SF): 1, 10, 30, 100, 300, 1000, 3000 which correspond to the nominal database size in GB. For example, SF 10 means a database with a size of approximately 10 GB. Tools for generating data and queries for the database are provided with the benchmark specification: DBGEN and QGEN. The use of these tools is strongly recommended and if not used, the mechanism for creating the data and queries must produce exactly the same output as those tools. TPC-H contains 22 well defined queries. Our DSS workload uses 17 out of the 22 queries. Sequential execution of these queries produce a query stream. The used queries are: q1-q16 and q19. Refresh functions perform database update between tests. The workload consists of three parts: Load Test, Power Test and Throughput Test. The Load Test just measures the time it takes for the system under test to build the database. The Power Test performs a pair of refresh functions and queries submitted in a single query stream. The test is performed sequentially (i.e., no concurrency). The Throughput Test performs multiple concurrent query streams and refresh functions.

4. Experiments

4.1. Single Class Model

The experiment consisted on running the workloads in a Data Center Server-SAN attached system using a same volume and DBMS. The implementation was done using a MYSQL 5.0 DBMS [7], C++ and MYSQL++ [8] tools. The experimental setup for this experiment depicted in Figure 2. The server is a DELL PowerEdge 1950 server with dual processor, dual-core, 2.0 GHz Intel Xeon EM64T processors with 8GB memory configuration connected to an EMC CLARiiON CX300 storage array via fibre channel with Red Hat Enterprise 4 (Linux 2.6.9-34.ELsmp, 64-bit kernel). The fibre channel connection is made through a using a McData Sphereon 1440 fibre channel switch. The server has a dedicated LUN of 1 TB consisting of 7 HDDs with a RAID 5 configuration. OLTP and DSS

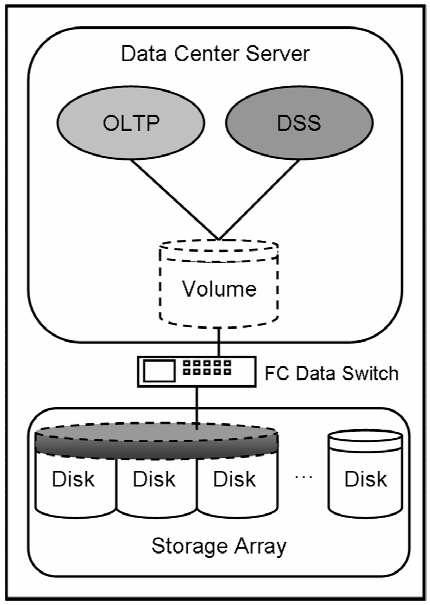


Figure 2. Experimental Setup

Table 3. OLTP workload description

Warehouse Number	Number of Terminals
10	100

implementations have the configuration shown in Tables 3 and 4. The databases were previously created and loaded. We decided to keep the applications size small to capture transaction behavior without stressing the system, specially when running concurrently. In the case of the DSS system we ran a complete Power and Throughput test sequence.
We collected disk I/O information of execution using linux command iostat [9] and EMC Navisphere CLI application [10]. These commands return information about the SAN treating it as a large disk. In this paper we will follow the same approach. We selected our model input data after the first 10 minutes of execution to allow time for the system to arrive at a near to steady-state condition. From this time interval, We will obtain the input parameters for our model such as average service time and arrival rate and compare the output of the model with the real results measured by

Table 4. DSS workload description

Scale Factor	Throughput Streams
1	10

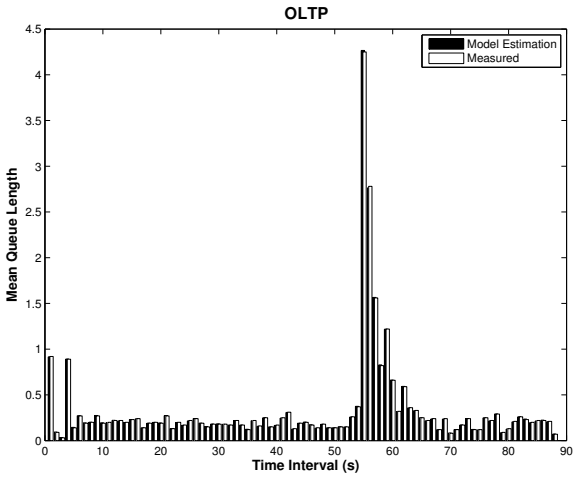


Figure 3. OLTP Queue Length Estimated by model vs Measured

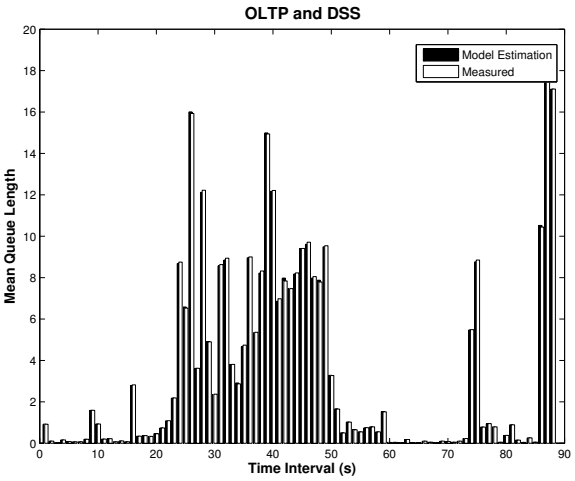


Figure 5. OLTP and DSS Queue Length Estimated by model vs Measured

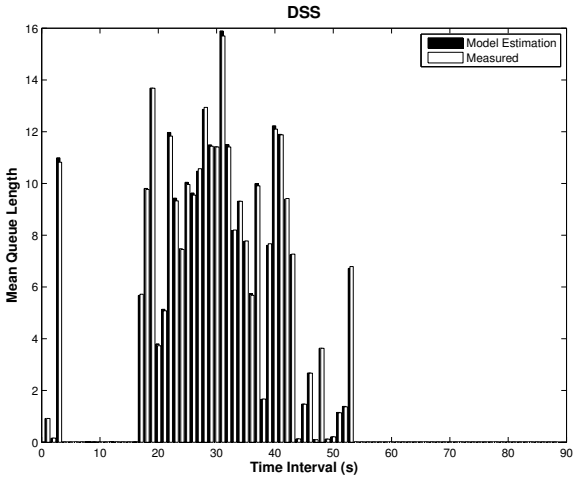


Figure 4. DSS Queue Length Estimated by model vs Measured

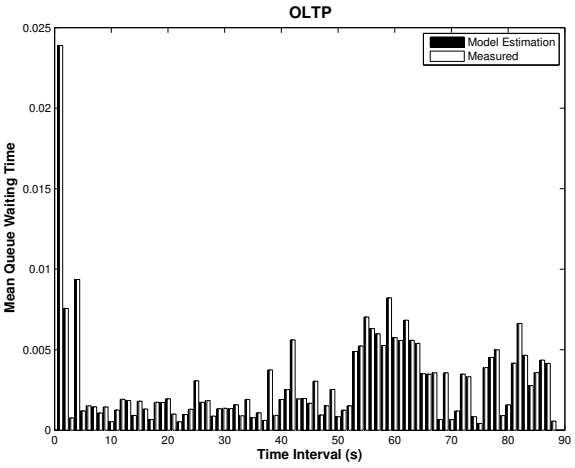


Figure 6. OLTP Queue Waiting Time Estimated vs Measured

these tools. The procedure is as follows:

1. Select a steady-state region in our simulation time (middle). We will collect the average service times, arrival rates, and response times.

2. Construct our model inputs, with these values and with the estimation of the coefficient of variance C_S^2.

3. Estimate our model outputs, the average queue length Q_L and average waiting time T_W.

We will display the estimated values of Q_L and T_W using our model and their respective measured results obtained during simulation.

Table 5 shows basic characteristics of the workloads. From these results, we can see that the multiple application (OLTP & DSS) is dominated mostly by the behavior of the DSS workload which has an I/O activity much larger than OLTP. This I/O activity is due to writes performed in the update functions. We can see from Figures 3 to 5 that the $M/G/1$ model predicts the average queue length with high accuracy. The error between measured results and the model estimates range from 0.01% to 4.26%. In the case of

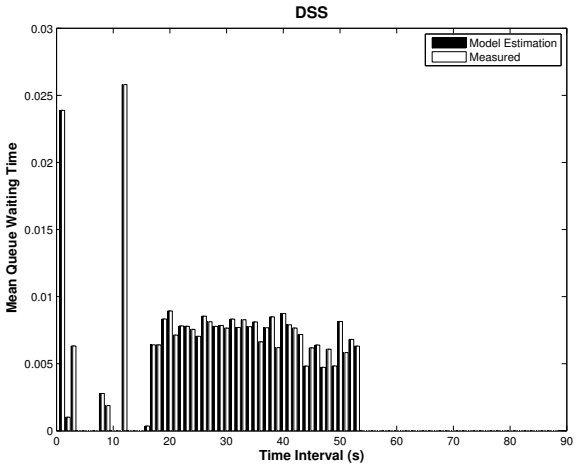


Figure 7. DSS Queue Waiting Time Estimated vs Measured

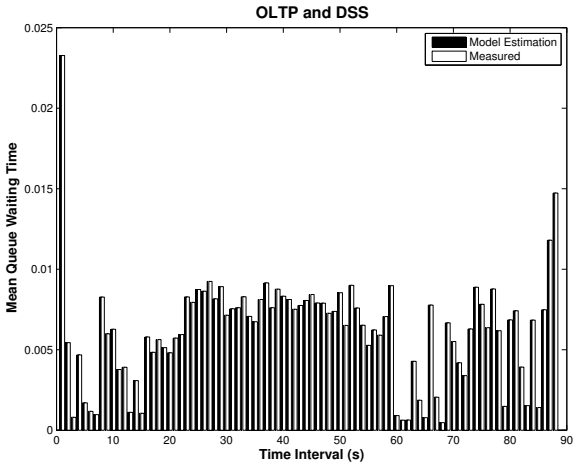


Figure 8. OLTP and DSS concurrent Queue Waiting Time Estimated vs Measured

the average waiting time, (Figures 6 to 8) the behavior is the same with error ranges from 0.03 to 16%. The 16% error is from DSS and is associated with the fact that few disk I/Os were generated with it. In the case of the concurrent the maximum error was 3.42% and OLTP error was 4.26%. The previous results give us confidence that the model has captured the behavior of the system. In the case of the multiple application results (Figures 5 and 8), we only have aggregated values at this moment, so we would have to model it with a single class model. In the next section we will analyze this model and how it behaves in terms of modifying these components to create a two class model.

Table 5. Workloads I/O Characteristics

Workload	R/W ratio	% reads	% writes	total IOps
OLTP	0.461	0.316	0.684	4,254.2
DSS	0.001	0.001	0.999	35,392
OLTP & DSS	0.005	0.005	0.995	34,309

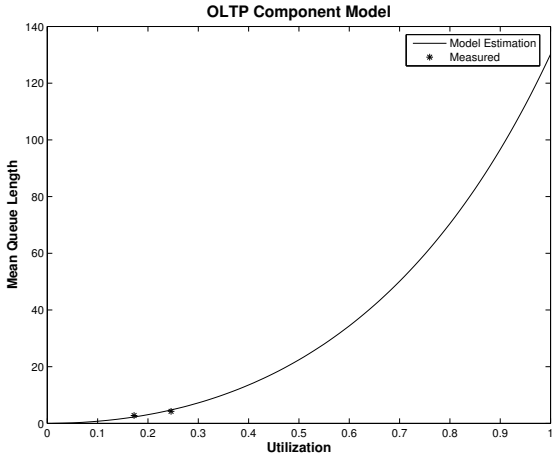
4.2. Two Class Model

The application of the previous models following a single class model approach produced excellent results. However, to fully understand the contribution of each workload, the multiple application experiment had to be modeled using a two class approach. Each input parameter representing the load of the classes were taken from the measured values (total). They were split in a manner such that the aggregation of both components resulted in the total measured values, that is, $\lambda = \lambda_{OLTP} + \lambda_{DSS}$, $\rho = \rho_{OLTP} + \rho_{DSS}$ and so on. We selected λ_{OLTP} to be around 20% the size of λ in order of satisfying the proportion we observed from Table 5. The service times ($T_{S_{OLTP}}$ and $T_{S_{DSS}}$) of this two class model are independent of each other. For simplicity we chose both to have the same average value, which was the measured one. We computed $C^2_{S_{OLTP}}$ and $C^2_{S_{DSS}}$. The output of our model will be the sum of the two components, that is: $Q_L = Q_{L_{OLTP}} + Q_{L_{DSS}}$. Since time values have the same average value, $T_{W_{OLTP}} = T_{W_{DSS}} = T_W$. The output of this model is the same as shown in figures 5 and 8 so we are confident that this approach can be used to analyze each component independently.

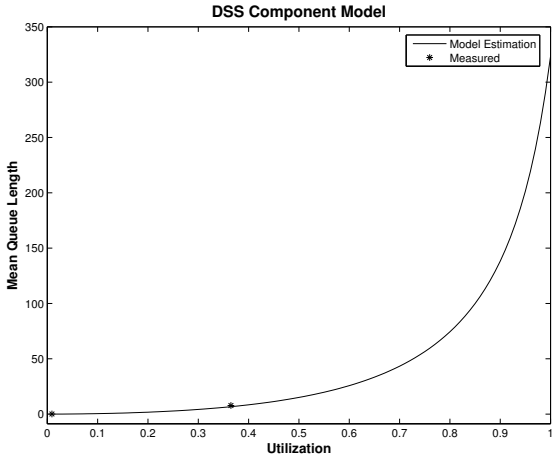
5. Analysis of Results

After validating our model, we can then proceed to analyze it. In this case we have selected a time interval inside the steady state region. The first step was to select similar measured values of T_S and their corresponding calculated $C^2_{S_{OLTP}}$ and $C^2_{S_{DSS}}$. Q_L values were calculated using the selected (T_S,C^2_S) pairs and varying ρ. ρ values were selected from 0-1 range so that we can see system behavior when modifying the load of each application component. The two class model was the only model used for this analysis. We split the model into the two components and draw the graphs of ρ vs Q_L. We also provide results from both components as a whole.

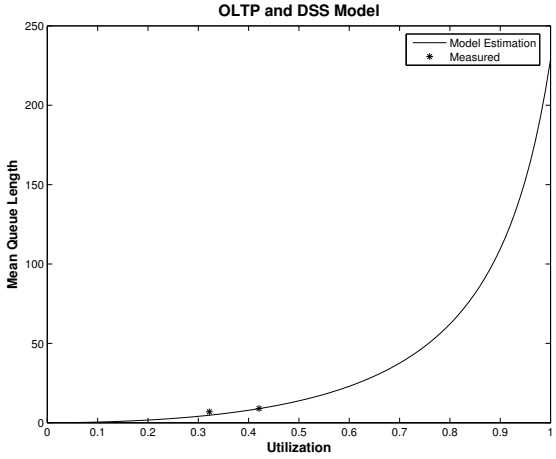
Figure 9 shows the obtained results. In (a) there is the OLTP component on the model. In this graph we let $\lambda_{OLTP} >> \lambda_{DSS}$, specifically $\lambda_{OLTP} = 0.9 * \lambda$. The graph shows a saturation point at around $\rho = 0.6, Q_L = 30$. Looking at the measured data for the single OLTP execution, we found two values with the same T_S for this anal-



(a) $\lambda_{OLTP} >> \lambda_{DSS}$

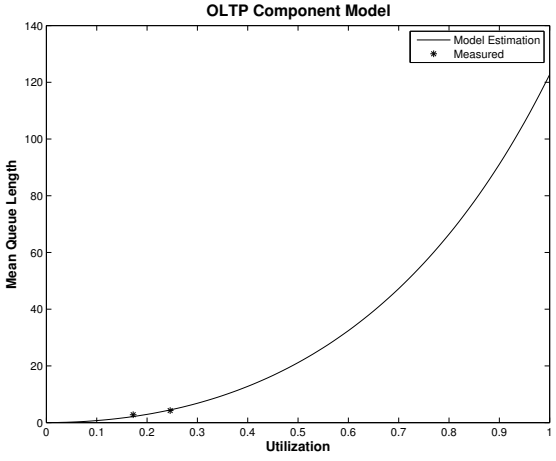


(b) $\lambda_{DSS} >> \lambda_{OLTP}$

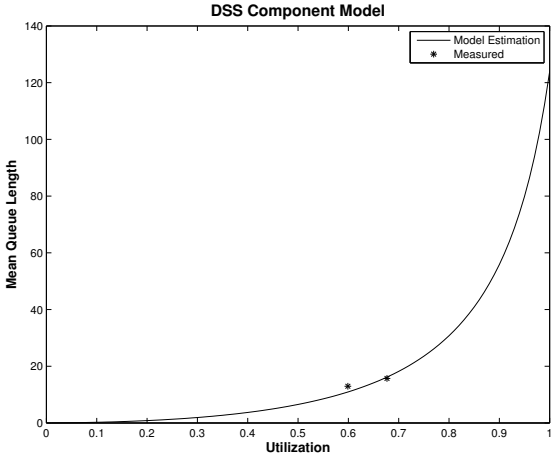


(c) Unmodified Model

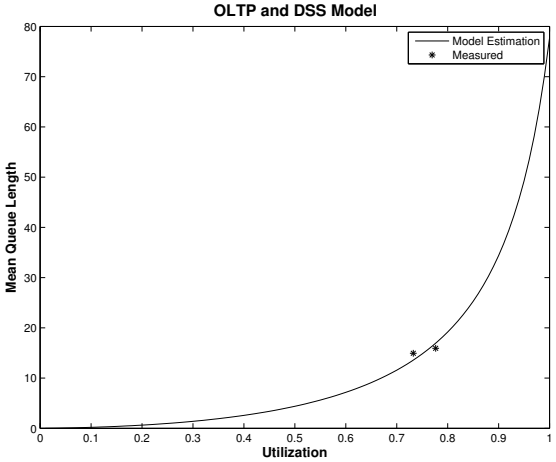
Figure 9. M/G/1 Two-Class Model: T_S = 0.40 ms



(a) $\lambda_{OLTP} >> \lambda_{DSS}$



(b) $\lambda_{DSS} >> \lambda_{OLTP}$



(c) Unmodified Model

Figure 10. M/G/1 Two-Class Model: T_S = 0.50 ms

ysis. When the values for the measured ρ and Q_L are plot, they fall inside the model graph. Figure 9 (b) shows the same type of plot associated with the DSS component of the two class $M/G/1$ model. In this case, we let $\lambda_{DSS} >> \lambda_{OLTP}$, with $\lambda_{DSS} = 0.8 * \lambda$. This component behaves very different than the OLTP component. The saturation point is located at around $\rho = 0.8$, $Q_L = 50$, which shows that this component saturates at a later utilization than that of the OLTP. Data values from the DSS single execution were also found and superimposed in the graph. We can see that the values from the DSS single experiments fit in the model's ρ vs Q_L curve. Finally, in (c) we have graphed both components of the model with values from the measured OLTP-DSS simulation. These values are also inside the model path as expected.

Figure 10 shows the same experiments but with different a value of T_S. The purpose of this figure is to analyze how the model behaves when the overall utilization is higher, specifically larger than 50%. We can see that also in this case, the model provides results that are close to the measured data. When analyzing both figures, we note that the combined two class model shows a behavior similar to that of DSS which is consistent with the measured results summarized in Table 5. This makes sense since DSS is the larger contributor in the mix. This analysis suggests that the two class model can be used to predict system behavior with certain similarity of actual measured data. In other words, the analysis shows that there is a contribution of each workload to the total mix and that these contributions can be separated while maintaining their respective behavioral attributes.

6. Related Work

The study of Storage Area Networks has gained popularity as this technology has emerged as one of the most commonly deployed storage solutions. Our work implements a queue model and uses it to model a particular system using obtained measurements. We are interested in modeling multiple complex database applications running concurrently.

Most of the previous efforts for SAN research have been in introducing theoretical models. Chao, Li-zhu and Chunxiao [11] proposed the use of an $M/M/Q$ model for Performance simulation of SANs. The authors construct a software simulator and show how this model is general enough to be fitted to different SAN systems. Their work shows an error of around 10%. Zhu, Zhu and Xiong [12], presented a Queueing Network Model to analyze SAN structures. They create a theoretical model that details all mayor components of a SAN as a network of queues. Besides the theoretical model, the authors provide analysis for the host server, fibre channel network, disk array subsystem, fibre channel protocol and disk units. Molero et al [13] developed a tool for the design and evaluation of fibre channel SANs. This tool is based on the CSIM C/C++ [14] extension to create simulations.

Menasc, Pentakalos and Yesha [15] present an analytical Queueing Network Model for mass storage systems. They consider both host attached and network attached devices. They provide a modified Mean Value Analysis (MVA) algorithm that can handle multiple classes of requests for a striped RAID array. Their workloads were a set of ftp get and put for a ten day period. Nijim, Xie and Qin [16] propose the use of an analytical model to model the performance of a self-managing computer systems under mixed workloads conditions. They perform a trace driven simulation where their workload mix consisted of one cpu intensive and one I/O intensive. Urgaonkar et al [17] present an analytical Queueing Model for multi-tier Internet applications. Their model predicted average response times within the 95% confidence intervals of the observed average response times.

7. Conclusions

In this paper we have shown results from modeling a single server connected to a SAN. We used a single $M/G/1$ Queueing Model. The experiments deal with characterizing the behavior of multiple workloads running in the same system. The results show how the model manages to represent the system of interest. This model is able of generating output parameters that yield low error when compared to experimental data. We learned that the two class $M/G/1$ model is able of predicting system behavior that is actually similar to measured data. The model can be used to anticipate the impact of growth in the arrival rate on the behavior of the system. In the case of adding more workloads to the mix, the model has the potential of predicting the system behavior once the new mix has been characterized.

As future work, we plan to extend these experiments by both increasing the workload number and size. We are also interested on quantifying the impact of each instance independently as a component of the whole disk utilization. This will require us to extend the model to a multi (more than two) class model. This would help us understand better the overall performance of the system given a particular set of desired applications to be run concurrently. The use of Queueing Network Models to characterize the complete system in terms of its components components as the fibre-channel switch, the storage processors and host bus adapters is the long term goal. This approach will provide an even greater level of understanding of all the characteristics this server-SAN system has.

8. Acknowledgments

This work was supported in part by an NSF Major Research Instrumentation Grant (Award Number MRI-0619616), the Institute for Complex Scientific Software, and from the Gordon-CenSSIS, the Bernard M. Gordon Center for Subsurface Sensing and Imaging Systems, under the Engineering Research Centers Program of the National Science Foundation (Award Number EEC-9986821).

References

[1] L. Huang,G. Peng and T.C. Chiueh. Multi-dimensional storage virtualization. In *SIGMETRICS 2004/PERFORMANCE 2004: Proceedings of the joint international conference on Measurement and modeling of computer systems*, pages 14–24, ACM Press, New York, NY, USA, 2004.

[2] Transaction Processing Council, TPC-C and TPC-H benchmarks specification (URL), `http://www.tpc.org`; accessed August 12, 2007.

[3] L. Kleinrock. Queueing Systems. Volume I: Theory. Wiley, New York, 1975.

[4] D. Bertsekas and R. Gallager. Data Networks, second edition. Prentice Hall, New Jersey, 1992.

[5] G. Bolch, S. Greiner, H. deMeer and K.S. Trivedi. Queueing Networks and Markov Chains. Wiley, New York, 1998.

[6] E.D. Lazowska, J. Zahorjan, G.S. Graham and K.C. Sevcik. Quantitative System Performance: Computer System Analysis Using Queueing Network Models. Prentice Hall, New Jersey, 1984.

[7] MYSQL AB, MYSQL Database (URL), `http://www.mysql.com`; accessed August 12, 2007.

[8] TangenSoft.net, MYSQL++ library (URL), `http://tangentsoft.net/mysql++/`; accessed August 12, 2007.

[9] S. Godard, Iostat man file, systat utilities home page (URL), `http://perso.wanadoo.fr/sebastien_godard/`; accessed November 23, 2007.

[10] EMC^2 Corporation, Navisphere SAN Management Suite (URL), `http://www.emc.com/products/storage_management/navisphere.jsp`; accessed July 15, 2007.

[11] L. Chao, Z. Li-zhu, X. Chun-xiao. Using MMQ Model for Performance Simulation of Storage Area Network. In *ICDE 2005: Proceedings of the 21st International Conference on Data Engineering*, IEEE Computer Society, Washington, DC, USA, 2005.

[12] Y. Zhu, S. Zhu and H. Xiong. Performance Analysis and Testing of the Storage Area Network. In *MSST 2002: 19th IEEE Symposium on Mass Storage Systems and Technologies*, IEEE, Maryland, USA, 2002.

[13] X. Molero, F. Silla, V. Santoja and J. Duato. A Tool for the Design and Evaluation of Fibre Channel Storage Area Networks. In *SS 2001: Proceedings of the 34th Annual Simulation Symposium*, IEEE Computer Society, Washington, DC, USA, 2001.

[14] Lockheed Martin Advanced Technologies Laboratory, The CSIM Simulator (URL), `http://www.csim.com`; accessed November 18, 2007.

[15] D.A. Menasc, O.I. Pentakalos and Y. Yesha, An Analytic Model of HierarchicalMass Storage Systems with Network-Attached Storage Devices, In Proceedings of the 1996 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, ACM Press, Philadelphia, May 1996.

[16] M. Nijim, T. Xie and X. Qin, Integrating a PerformanceModel in Self-Managing Computer Systems under Mixed Workload Conditions, In Proceedings of the IEEE International Conference on Information Reuse and Integration,IEEE, August, 2005.

[17] B. Urgaonkar, G. Pacifici, P. Shenoy,M. Spreitzer and A. Tantawi, Analytic modeling of multitier Internet applications, In ACM Transactions on the Web (TWEB), Volume 1, Issue 1, ACM Press, New York, May, 2007.