Analytic Performance Models of Robotic Storage Libraries: Status Update

Theodore Johnson
johnsont@research.att.com
AT&T Laboratories – Research
Florham Park, NJ 07932
tel +1-973-360-8779
fax +1-973-360-8050

Abstract

Large tertiary storage systems tend to be expensive, so performance models are needed for sizing and for performance optimization. Several researchers (including ourselves) have attempted to supply these models in recent years. In this paper we report on our progress in the development of a previously published model. The improvements we have made include a more accurate model of the robot arm, experimental validation of the model, and the integration of the model into PC analysis tools through a Perl interface.

1 Introduction

In spite of rapidly declining on-line storage prices, many applications, such as scientific archives, scientific databases, data warehouses, digital libraries, and multimedia servers create and serve such large volumes of data that the use of tertiary storage is required. Large tertiary storage systems tend to be expensive, and tend to have unusual performance characteristics. Performance models are needed for sizing and performance optimization studies.

The need for performance models has motivated considerable recent research. Johnson [3, 4], and Menasce, Pentakalos, Yesha, and Halem [7, 8] have developed analytical performance models of tertiary storage systems. Gibson and Miller [1] are developing a configurable mass storage simulator. The NASA Goddard Mass Storage Testing Laboratory [9] is developing a benchmark suite to test and rate HSM solutions.

In [3, 4], we presented a detailed queuing model of a robotic storage library, incorporating the interaction of robot arm, batch arrivals, and multiple file loads from a tape. Since then, we have continued to work on improving the accuracy and usefulness of the model. In particular,

- A weakness of previous tertiary storage models was the lack of detailed information about the performance of model components. We have performed an extensive performance characterization study of common tertiary storage components [6]. We have used this information to make our analytical model more realistic.
- We have refined our model of the robot arm, increasing its range of accuracy.
- We have validated our model using experimental measurements of a Storagetek 9710 with four DLT 4000 drives.
- We have developed an interface between our model and common PC data analysis tools, using widely available public domain software such as Perl.

In this paper, we report on the last three refinements (please refer to our paper [6] in these proceedings for information regarding tape drive performance characterizations).

2 Robot Arm Modeling

In the model reported in [3, 4], we describe how the interaction between queuing for service from the robot arm of robotic storage library and queueing for service from the tape drives can be modeled with an iterative computation.

A *job* in our model represents a user's request for data. Our analysis of usage patterns [2, 5] indicated that a typical request is for a batch of files, possibly distributed over multiple media. In our model, a job splits into a batch of requests, where each request represents the loading of data from a media. The time to serve a request is the sum of the media fetch time, mount time, seek times, and file transfer times. Each request queues for service from one of the media drives, and the job is finished when all of the requests in the batch are finished. The media fetch time depends on the drive utilization, and vice versa, leading to the iterative solution. The robot arm is modeled as a $M/G/1$ queue with an infinite customer population and batch arrivals. While the actual customer population is finite, for light loads the approximation is reasonably accurate.

In the process of performing an experimental validation, we found that the robot arm could easily be made to have a high utilization. The infinite customer model overestimated queuing delays at the robot arm, decreasing the accuracy of the model. To fix the problem, we developed a finite customer population model. In this section, we briefly describe the approach.

The software we use to manage the robotic storage library has a volume manager that does not rewind tapes after use, nor does it return the tapes to the shelf. In addition, the volume manager is single threaded, so only one tape can be changed at a time[1]. As a result, the

[1]If we disable the "fast load" setting on the Storagetek 9710, the robot does not return a ready status until the drives are ready (in case the tape handling software cannot detect when the drive is ready). This setting also results in very long robot service times.

time spent by a request in service by the volume manager can be represented by the robot arm server. The service time of the volume manager includes rewinding the tape in the selected drive, unmounting it, returning the old tape to the shelf, fetching the new tape, and mounting it.

Let c be the number of tape drives in the robotic storage library. If b of the c drives are busy, then the the maximum number of tape load requests pending service by the volume manager is $c - b$. If there are $q < b - c$ load requests pending service by the volume manager, then a new load request must wait for the volume manager to finish the existing q requests.

The model described in [3, 4] assumes that a user fetch request translates into multiple media fetch requests As a result, media load requests also come in batches, but the batch size cannot be larger than the number of idle drives, $i = c - b - q$. Let B be the random variable representing the number of media loads generated by a user fetch request. Let $p_j = \Pr[B = j]$ be the distribution of B. Then if i drives are idle, the size of the batch arrival at the volume manager queue is the random variable B_i with the distribution $p_{i,j} = \Pr[B_i = j]$ defined by

$$p_{i,j} = \begin{array}{cc} p_j & j < i \\ \sum_{k=i}^{\infty} p_k & j = i \\ 0 & j > i \end{array}$$

We can describe the state of the volume manager by specifying the number of busy, queued, and idle drives. Since $b + i + q = c$ and c is constant, we only need to specify two of the three parameters to describe the state. It turns out to be easiest to specify the state of the volume manager with (b, i). The volume manager can be in any state (b, i) such that $b \geq 0$, $i \geq 0$, and $i + b \leq c$.

Next, we specify how the volume manager can change states. The transitions are given in Table 1. In this table, E_{tr} is the expected service time of the volume manager, E_{dr} is the expected service time of drive to fetch the requested data, and p_{qd} is the probability that if all drives are busy and a drive finishes service, there is pending fetch request for the drive to service.

The states and the transition function of the volume manager model define a finite Markov chain model, which we can solve for the steady state behavior. The model has $O(c^2)$ states, and c is small. The robot arm model can be extended to handle deterministic robot arm service times [6] (by using an embedded Markov chain model) and separate media fetch and return requests. We will detail these models in a full paper.

cause	from	to	rate	condition
robot completes service	(b,i)	$(b+1,i)$	$1/E_{tr}$	$b < c-i$
drive completes service no queued requests	(b,i)	$(b-1,i+1)$	b/E_{dr}	$i>0, b>0$
drive completes service no queued requests	(b,i)	$(b-1,i+1)$	$(1-p_{qd})b/E_{dr}$	$i=0, b>0$
drive completes service queued requests	(b,i)	$(b-1,i)$	$p_{qd}b/E_{dr}$	$i=0, b>0$
job arrival	(b,i)	$(b,i-j)$	$p_{i,j}\lambda$	$j \leq i$

Table 1: State transitions for the robot arm (volume manager).

3 Experimental validation study

We had access to a Storagetek 9710 with four DLT 4000 tape drives. As discussed in the previous section, the volume manager of our tape management software is single threaded, requiring a modification in the definition of the robot service time and the drive service time. We measured the Storagetek 9710 and the DLT 4000 to obtain service time parameters (please see [6] for details on the measurements).

We submitted synthetic jobs to the robotic storage library. A controller script generated requests by waiting for an exponentially distributed length of time and then creating a batch request. A batch request was generated by forking off k processes each of which requested the load of a media, and the fetch of files from the media. The batch size is chosen by sampling an exponentially distributed random variable, while the number of files fetched from a tape has a uniform distribution. Because of the limited number of test tapes (40), we limited the batch size to 15 media. The size of a file fetched from tape can be 1 Mbyte, 10 Mbytes, 50 Mbytes, 200 Mbytes, or 1 Gbyte with a 30%, 30%, 20%, 15%, and 5% chance, respectively, representing a wide range of file sizes. We note that the wide range of file sizes and the emphasis on small files makes obtaining an accurate model more of a challenge.

We varied the arrival rate, the average number of media fetched, and the average number of files fetched per media, and measured the batch waiting time and the batch service time (i.e., by using the waitpid system call). Each experiment was terminated after 50 jobs completed. We assumed that seeks required an average of 75 seconds, the robotic service time is 9 seconds, the mount time is 40 seconds, the rewind and unmount time is 100 seconds, and the transfer rate is 1.5 Mbytes/sec, based on our benchmark measurements [6]. Table 2 lists the results.

The response time predictions are usually within 20% of the observed response time, and in all cases is within 30%. In some cases the predicted response time is quire close to the observed. This good agreement (and also the cases of poor prediction) is due to random chance, as only one sample path was observed. However the general trend is that the model

avg. time between arrivals	avg. media per job	avg. files per media	number of drives	experimental response time	analytical response time	percent difference
1000 sec.	2	5	3	3438 sec.	3571 sec.	3.8%
750	2	3	4	1473	1695	15
1000	2	3	4	1283	1297	1.1
1000	4	3	4	2813	3273	16
1400	3	3	3	2755	2247	18
1000	2	3	3	1293	1340	3.6
1200	4	3	4	3432	2473	28

Table 2: Experimental and analytical predictions of job service times.

gives reasonably accurate predictions. We note that the device utilizations were high, with a drive utilization ranging from 49% to 70%, and a robot arm (volume manager) utilization ranging from 24% to 38%. Accurate response time predictions are difficult when device utilizations are high.

Although we generated a synthetic workload that matches the modeled workload, the processing of the workload was performed by the robotic storage library and its driving software. Therefore we conclude that our model of the mechanics of the robotic storage library are accurate.

4 Interfaces

We have ported the model solver to the Win 95 platform. The model solver is written in ANSI C, so POSIX compliant C development environments (for example, DJGPP, which is available for free) can compile the model and generate executable code.

The Perl scripting language is an excellent tool for driving an external program, parsing the results, and generating reports. Perl is also available for the Win 95 platform for free. Recent versions of Perl (e.g., Perl 5) support OLE calls. As a result it is easy to write a Perl program that makes multiple calls to the model solver, collates the results, passes a table to a spreadsheet package, and causes the spreadsheet to plot the results. We have developed some sample scripts.

5 Conclusions

Understanding the performance of tertiary storage devices is a difficult task, but is necessary for the planning and optimization of large data systems. In this paper, we present our progress in developing an analytical model of a robotic storage library. By developing a

more accurate model of the robot arm and by taking measurements of a Storagetek 9710 with four DLT 4000 tape drives, we were able to accurately model the system. To simplify the use of the model, we have developed interfaces to tools available on a PC platform.

Acknowledgements

We'd like to thank the Session Chair, Jean-Jacques Bedet, for his comments on a draft of this paper.

References

[1] T. Gibson and E. Miller. Long-term file activity patterns in a unix workstation environment. In *Proc. NASA Godddard Conf. on Mass Storage Systems and Technologies*, 1998.

[2] T. Johnson. Analysis of the request patterns to the nssdc on-line archive. In *NASA Goddard Conf. on Mass Storage Systems and Technologies*, pages 367–382, 1995.

[3] T. Johnson. An analytical performance model of robotic storage libraries. In *Performance '96*, pages 231–252, 1996.

[4] T. Johnson. Queuing models of tertiary storage. In *NASA Goddard Conf. on Mass Storage Systems and Technologies*, pages 529–552, 1996.

[5] T. Johnson and J. Bedet. Analysis of the access patterns at the GSFC Distributed Active Archive Center. In *NASA Goddard Conf. on Mass Storage Systems and Technologies*, pages 153–178, 1996.

[6] T. Johnson and E. Miller. Performance measurements of robotic storage libraries. In *Proc. IEEE Conf. on Mass Storage Systems / NASA Goddard Conf. on Mass Storage Systems and Technologies*, 1998.

[7] D. Menasce, O. Pentakalos, and Y. Yesha. An analytical model of hierarchical mass storage systems with network attached storage devices. In *SIGMETRICS 96*, pages 180–189, 1996.

[8] O. Pentakalos, D. Menasce, M. Halem, and Y. Yesha. Analytical performance modeling of mass storage systems. In *Proc. 14th IEEE Mass Storage Systems Symp.*, 1995.

[9] R. Venkataraman, J. Williams, D. Michaud, P. Hariharan, B. Kobler, J. Behnke, and B. Peavey. The mass storage testing laboratory at GSFC. In *Proc. NASA Godddard Conf. on Mass Storage Systems and Technologies*, 1998.