A QoS-based Disk Subsystem *

Athena Vakali
Department of Informatics
Aristotle University
54006 Thessaloniki, Greece
avakali@csd.auth.gr

Constantina Stupa
Department of Informatics
Aristotle University
54006 Thessaloniki, Greece
kstupa@csd.auth.gr

Abstract

QoS has been proposed in storage subsystem management towards effective disk space utilization and request servicing. We present a QoS based storage model for effective user negotiation in terms of scheduling and the number of storage devices. A simulation model is developed based on an available disk simulator which is experimented under artificial request workload towards better system's responsiveness, performance and functionality. Certain remarks and conclusions are raised with respect to the simulated scheduling algorithms and the capacity available to the client's environment.

1 Introduction

The increasing need of more efficient and effective storage configurations has become more imperative due to the wide spread of multimedia data which demand great storage capacities together with synchronization and appropriate retrieval. The term "Quality Of Service" (*QoS*) was introduced to describe certain tecnical characteristics (mainly in communications technology) such as performance, speed and reliability. An overall definition in relation to QoS with multimedia applications is given in [11] : *Quality of Service represents the set of those quantitative and qualitative characteristics of a distributed multimedia system necessary to achieve the required functionality of an application.*

Attribute managed storage is discussed in [1, 5] where the development of a new storage system is proposed such that mapping of virtual to physical storage devices is introduced with quality of service guarantees. Here, we consider the case of attribute managed storage in order to effectively manage storage resources with respect to the most important characteristics such as scheduling and multiple disk configurations.

The remainder of the paper is organized as follows : the next section introduces the storage model whereas Section 3 describes the QoS negotiation scheme and the attribute managed storage process. In Section 4 the disk simulator used is described and the the experimentation details and results are given. Finally, conclusions and further research topics are discussed in Section 5.

*This work is supported by the National Science Foundation under grants 9972883-EIA, 9974255-ISS and 9983249-EIA; grants from HP, IBM, Intel, Telcordia and CERIAS; and Indiana 21st Century grant from the State of Indiana.

2 The Storage Model

Parameter	Meaning
T_{seek}	Seek time
$T_{transfer}$	Transfer time
$T_{overhead}$	Overhead time
T_{switch}	Switch time
$T_{rotatioanl}$	Rotational latency
T_{rot}	Rotation time for a disk
s_t	Size of a track
R_t	Data transfer rate within a track

Figure 1: QoS parameters in storage subsystems.

Storage subsystems performance and functionality depends on the certain disk's topology configuration and characteristics. Usually disk drive performance is measured by how fast they can satisfy a user's request. The following parameters have been the most important ones to characterize disk's performance :

- *Seek Time* is the time to move the head from its current cylinder to the cylinder specified by the next request. The most typical equation that defines the seek time for a disk head "travel" of s cylinders under a specific HP disk model [7] is :

$$T_{seek} = \begin{cases} 3.24 + 0.40\sqrt{s} & s \leq 383 \\ 8.20 + 0.0075s & s > 383 \end{cases} \quad (1)$$

- *Rotational Delay* or *Rotational latency* is the time it takes for the target sector to rotate under the head, expected rotational latency is (T_{rot}) :

$$T_{rotational} = \frac{1}{2} T_{rot} \tag{2}$$

- *Data Transfer Time* which depends on data rate and transfer size.The most typical formula for the data trasfer time is:

$$T_{transfer} = \frac{size_{transfer}}{R} \tag{3}$$

- *Average Request servicing Time* is the average time to service a random request as expressed by :

$$T_{overhead} + T_{seek} + T_{rotational} + T_{transfer} + T_{switch} \tag{4}$$

where $T_{overhead}$ is the time for the disk drive's microprocessor and electronics to process and handle an I/O request and the T_{switch} is the overall switch time for the disk to switch from one surface of the disk to the other.

3 QoS negotiation and attribute managed storage

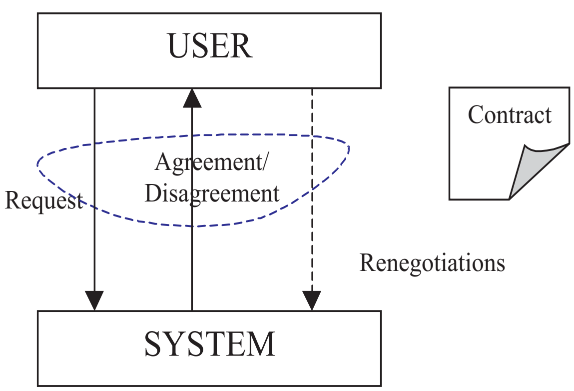

Figure 2: QoS contract negotiation process.

Here, we introduce a QoS based storage model which focused on the specification of certain QoS parameters in relation to : disk modeling and disk simulators and disk scheduling. The proposed QoS-based storage system supports the assignment and managenment of the QoS parameters negotiation between the user and the storage system. This task is divided into three step negotiation cycle :

1. *Assesing* the QoS requirements in terms of user's demands in relation to performance, synchronization, cost e.t.c.

2. *Associating* these requirements with QoS parameters

3. *Negotiating* between user and storage system components to ensure that the system can meet the required parameters.

If the negotiation fails then the above cycle activities will be repeated until negotiation succeeds. Due to the flexibility of this task, both the user and the storage system can change QoS requirements during an application session. In this case we result in a renegotiation phase. Whatever the case the negotiation task can result in three different types of agreement: quaranteed, best-effort or stochastic. Figure 2 depicts the generalized structure of a QoS negotiation process. The storage subsystem should monitor continuously the request servicing process, maintain the agreed QoS values and apply correction mechanisms in order to restore the system to its required condition.

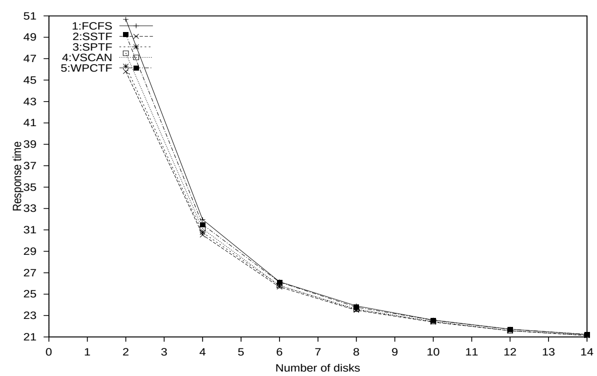

Figure 3: Response time for different scheduling algorithms under a 10000 request workload

4 Experimentation - Results

The proposed negotiation process is experimented under a developed disk simulator (*DiskSim*) and the experimentation involves request servicing under various scheduling policies and number of disk devices in the storage topology. DiskSim is an effective, strong disk system simulator implemented by G. Ganger, B. Worthington and Y. Patt [3]. The components that are emulated are: disks, controllers, buses and disk block caches. Their configuration is very detailed as it

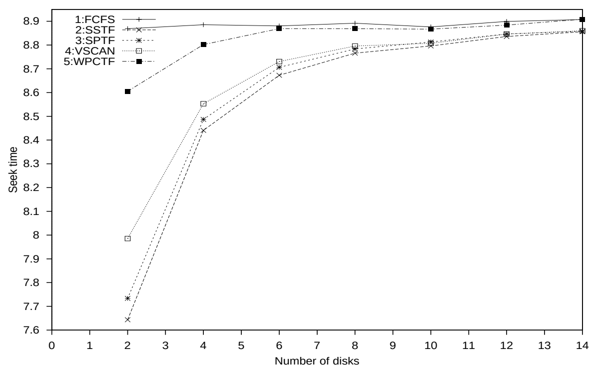

Figure 4: Seek time for different scheduling algorithms under a 10000 request workload

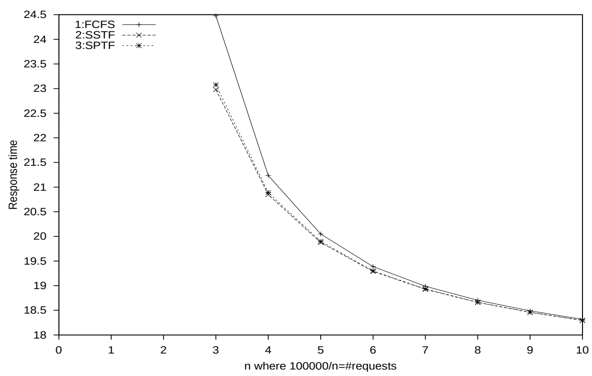

Figure 5: Response time for different scheduling algorithms under one disk drive subsystem.

involves a large number of parameters. We experiment with the adequate ones of them in order to assess the several data placement schemes, data topologies and scheduling policies. One feature of DiskSim is that it can work by using either traces or internally generated synthetic workload. Therefore, an analyzer is able to evaluate real workload by exploiting exisitng traces and to produce simulation results which will be the outcome of the service of synthetic workload. It has been proved [3] that the results are really similar, a fact that exacerbate this tool's power and capabilities. Disksim simulator involves a parameter file and a trace file if the synthetic generator is not activated in order to work. This file contains parameters for the most significant components of the storage subsystems which are the queue/scheduler subcomponents, the buses, the controllers, the disks and the caches. The most important parameter that is defined here is the scheduling policy. The user can choose between a large number of algorithms in order to schedule the service of the requests that arrive.

We have adjust the proposed negotiation to the DiskSim simulation environment, in a user friendly way. The negotiation parameters refer to Disk parameters, data organization and time variables. We have run several experimentation runs by specifying the following parameters :

- *The number of requests (in thousands)* : this parameter should be between 1 and 150 since a larger value would minimize the performance of the simulator.

- *Sheduling policy* : the user can choose among a variety of algorithms. DiskSim provides several algorithms to schedule requests in the waiting queue. The most indicative algorithms considered in our experimentation are : FCFS, SSTF, SPTF, WPCTF and VSCAN (details abou the algorithms are given in [3].

- *Disk ID* : the user may choose between 5 different disks : HP-C3323A, DEC-RZ26, HP-C2490A, HP-C2247A, HP-C2249A

- *Number of disks* : the user can specify the number of disks included in the storage subsystem. The values that can be used are 1 to 14 as the physical organization of the devices that we use cannot afford more disks connected to it.

We have used a trace file with 10000 requests and by changing the inter arrival time we managed to lighten the workload. Moreover, the workloads produced by the synthetic generator had the following characteristics:

- They consisted of 66% read requests and 34% write requests. If the percentage of write requests was greater then the results will show an increase in response time as these requests impose greater overhead.

- There were generated no sequential requests. A sequential request is one that its starting address is immediately after the last addressed accessed by the previous request.

- The size of the requests followed an exponential distribution with base value 0.0 and mean value 8.0

Indicative results of the experimentation are depicted in Figures 3 - 5. More specifically, Figure 3 has the curves of response time with respect to the number of disks when different scheduling algorithms are used under a 10000 requests workload. As depicted in this figure, FCFS algorithm shows the worst response time as it was expected. A really interesting feature is the sharp fall of response time from 2 disks to 4 disks. All of algorithms decrease their response times by about 50%. Thus, the more disks a subsystem has the more effective it is. Of course, after a certain number of disks the decrease in response time is not so great as to balance the increase in cost, the difference in response time for a subsystem with 12 disks and one with 14 disks is only 0.5 msec. Another remark is that the response times for the various scheduling algorithms are becoming equal as the number of disks is increased.

Figure 4 has the corresponding seek time for the above workload. Unlike response time, seek time increases as the number of disks becomes greater, except for the FCFS algorithm which remains almost the same. Again, the difference of the several algorithms is obvious when two disks are used. SSTF results in the smaller seek time and the algorithms that follow are SPTF and VSCAN. WPCTF algorithm shows quite large seek time which is not raising significantly as the number of disks increases.

Figure 5 depicts the experimentation when a trace file was used, the parameter file defined the characteristics described earlier and the subsystem used one disk drive of type HP–C2490A. The conclusions are similar to those discussed above. As the number of requests increases so does response time. In case of 33000 requests FCFS algorithm present a very poor performance in comparison to the other algorithms. The seek time for the above topology shows similar results to the seek time curve of the subsystem under a varying number of disks.

5 Conclusions - Future Work

QoS has been introduced towards an effective system's performance and utilization on a considered storage topology. A disk simulator was used to experiment under workloads of a varying number of requests and certain conclusions were discussed about the proposed QoS parameters specification. The negotiation cycles and the user demands specification has been proven quite beneficial with respect to the seek times as well as to the overall response times.

Further research should examine more complicated storage subsystems which could involve storage hierarchies such as caching, disks and tapes. It will be quite beneficial to employ QoS parameters at all storage levels in order to result in more effective user request servicing.

References

[1] E. Borowsky, R. Golding, A. Merchant, L. Schreier, E. Shriver, M. Spasojevic, and J. Wilkes : Using attribute-managed storage to achieve QoS, *HP Technical Report*, 1994.

[2] Huang-Jen Chen and Thomas D.C. Little : Storage Allocation Policies for Time-Dependent Multimedia-Data, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 8, NO. 5, October 1996.

[3] G. Ganger, B. Worthington and Y. Patt : The DiskSim Simulation Environment, *http://www.ece.cmu.edu/ ganger/disksim*, 1998.

[4] Gregory G. Ganger, Bruce L. Worthington, Robert Y. Hou, Yale N. Patt *Disk Arrays*, IEEE March 1994.

[5] R. Golding, E. Shriver, T. Sullian and J. Wilkes : Attribute-managed storage, *Workshop of Modeling and Specification of I/O MSIO*, San Antonio, TX, Oct. 1995.

[6] Y. Rompogiannakis, G. Nerjes, P. Muth, M. Paterakis, P. Triantafillou, G. Weikum : Disk Scheduling for Mixed-Media Workloads in a Multimedia Server, *ACM Multimedia*, Bristol U.K. 1998.

[7] Chris Ruemmler, John Wilkes : An Introduction to Disk Drive Modeling, *IEEE Computer* March 1994.

[8] Margo Seltzer, Peter Chen, John Ousterhout : Disk Scheduling Revisited, *USENIX Winter Technical Conference*, 1990.

[9] Spencer W. Ng : Advances in Disk Technology: Performance Issues, The IEEE Computer, 1998.

[10] Ralf Steinmetz : Multimedia file systems survey: approaches for continuous media disk scheduling, *ACM Computer Communications*, Vol. 16, No. 3, March 1995.

[11] Andreas Vogel, Brigitte Kerhervé, Gregor von Bochmann and Jan Gecsei : Distributed Multimedia and QoS: A Survey, *IEEE*, Summer 1995