Performance Analysis and Testing of the Storage Area Network

Yao-Long Zhu, Shu-Yu Zhu and Hui Xiong
Data Storage Institute
5 Engineering Drive 1
(off Kent Ridge Crescent, NUS), 117608, Singapore
Email: dsizhuyl@dsi.nus.edu.sg
Tel: +65 874-8695
FAX: +65 777-2053

Abstract

This paper is focused on the performance evaluation of storage network and storage system based on both theoretical modeling and experimental testing. The queueing network models of hard disk drive, disk array, and storage area network are presented. The effects of I/O request type, size, and system parameters on the SAN performance are analyzed. This study uses Fork/Join model in disk array performance analysis. We also present and analyze four schedule algorithms for FC-AL. The results show that the bottleneck of SAN performance is on the different node depending on different system I/O requests and applications. System performance for small sequential I/O requests are limited by the disk array controller and cache overhead, while big sequential I/O requests are restricted by the Fibre Channel (FC) network. As for the random I/O requests, the limitation lies on the performance of hard disk and disk array configuration. The theoretical results are compared with the test results, and found to be in agreement.

1. Introduction

The electronic marketplace, scientific data logging, and digital multimedia are aggressively pushing the demand for data storage. Storage Area Network (SAN) is an open storage architecture concept designed to meet the scalability and manageability of this astronomically growing storage requirement. Hence, we begin to see storage architecture and technology move away from Direct-Attached-Storage and towards Storage Area Network (SAN). It therefore becomes an interesting issue to be able optimize a SAN design and be bale to evaluate the performance for storage systems based on a SAN environment.

There are a number of prior works on the performance evaluation of the traditional storage system [1]. Most of the analytical models developed for disk array performance are either based on service time and distributions at the individual disk level, or focused on the SCSI bus connection [2][3]. However, efforts to analyze the total system performance of a Storage Area Network are still limited [4][5].

This paper uses the Queuing Network model for evaluating and analyzing the performance of the SAN and storage system. The modeling assumes a modular approach consisting of disk array subsystem module, storage device module, storage network module and host module. The theoretic results are compared with the experimental testing results. Performance bottlenecks for different I/O requests and applications are also analyzed and discussed for a typical SAN configuration.

2. SAN queuing network model

Typical SAN architecture consists of three main parts: host bus adaptor (HBA) on the hosts, storage network connection equipment (fabric switches and hubs), and storage system. There are numerous SAN architectures designed for different applications. For the purpose of this paper, a simple configuration of multiple hosts sharing a single storage system through a FC fabric switch is used. Figure 1 shows the queuing network model used for the SAN configuration and storage system architecture.

Service components included in the queue consist of multiple hosts, FC fabric network, disk array controller and cache (DACC), FC-AL connection, and disk units made up of disk controller and cache (DCC) and HDA. These components are presented as *service nodes,* and each is represented with a queuing model and different I/O request types and rates.

The key parameters considered in the queuing model are request rate and distribution, service time and distribution, and service disciple. The output parameters, considered in terms of the service node and entire SAN network's perspective, are queue waiting time, response time, queue depth, utilization and throughput. The model assumes that the workload for all service nodes has a Poisson arrival process, while the service disciples for all service nodes are assumed to be FCFS (first come first serve), unless the dedicated schedule algorithms are indicated.

2.1 System request and system response time

Within the host service node, only two critical functions are considered; that is the I/O request generation source and the HBA. Current servers use the system buses (such as PCI) to transfer the data between the HBA and the host memory. From the software viewpoint, I/O requests are initiated by the application through file system. The I/O requests are issued in the form of SCSI commands. SCSI middle layer and low-level

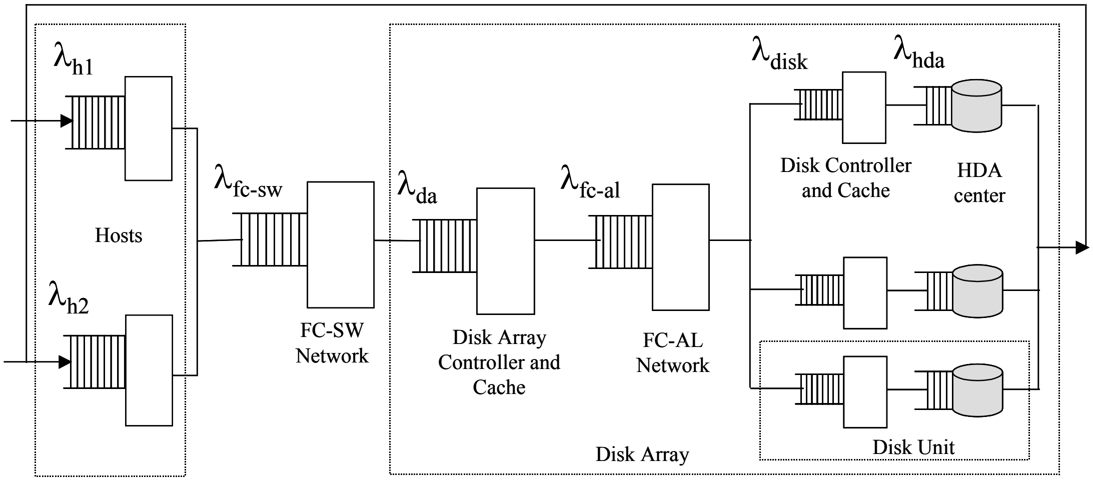

Figure 1. A queuing network model for storage system and storage network.

driver will then put the I/O request in a queue, which are passed to actual hardware via the Direct Memory Access (DMA). After the SCSI command has been processed by the SCSI device, acknowledgement is returned to the SCSI device driver. Therefore, the time from issuing an I/O request to receiving acknowledge information is defined as the system response time.

The system response time for I/O request type *'i'* is estimated as:

$$TR_i = \sum_{j \in S} p_{i,j} * TR_{i,j} \qquad \text{S=\{host, fcf, dacc, fcal, dcc, hda\}}$$

The mean response time of the system for all types of the I/O request is estimated as:

$$TR_j = \sum_{i \in R} p_{i,j} * TR_{i,j} \qquad \text{R=\{sequential/random read/write, destage, cmd, data)}$$

Where, $p_{i,j}$ is the probability of the request type '*i*' serviced by service node '*j*'. All notations including request types and input parameters are presented in Table 1.

2.2 Host service node

The first service node encountered by the system I/O request is the HBA. The service time the HBA is affected by the hardware overhead for command processing and data transfer time by DMA. Therefore, the response time for HBA to process an I/O request is estimated as [9],

$$TR_{host} = E[TS_{host}] + \frac{\lambda_{host} * E[TS_{host}^2]}{2(1 - \lambda_{host} * E[TS_{host}])}$$

2.3 Fibre Channel network

It was assumed there is only one target port (DACC) in the FC fabric (FCF) connection, and therefore the maximum FCF bandwidth for this evaluation is 100MB/s. The service

Table 1. Notation for SAN queue modeling

Service centers:
S = host, fcf, dacc, fcal, dcc, hda
Request type:
R = read, write, parity, destage, cmd, data,
Output parameters:
TR: system response time
TS: service time
TW: queueing waiting time
Input parameters:
N_h: host numbers
N_d: disk number to form disk array

Input parameters:
λ: I/O request rate
ρ: utilization $=\lambda * E[TS]$
μ: service rate $=1/E[TS]$
E[TS]: mean value of *TS*
$E[TS^2]$: the 2nd moment of *TS*
$P_{i,j}$: probability of *'i'* on *'j'*
t_{dt}: data transfer time
t_{hs}: head switch time
t_{la}: latency
t_{cs}: cylinder switch time
t_{seek}: seek time

time of the FCF consists of 3 parts: switch delay, physical distance transmission delay, and data transfer time. For long distances (~100km), the physical layer transmission latency will cause performance loss, as a result of the SCSI transmission mechanism requiring a “receiver-ready” acknowledgement in the write operation [6]. This is not an issue for short distances between the initiator and target connection (typically, only several meters in the lab environment and storage system). Therefore, the data transfer time is considered as the primary service time overhead.

System request rate from different hosts are combined in the FL-port of the FCF switch. The request rate of FCF and DACC is sum of the all host I/O request.

$$\lambda_{fcf} = \lambda_{dacc} = \sum \lambda_{hi} \qquad i = 1, \quad 2 \ldots \quad N_h$$

The queue waiting time of an I/O request induced by FCF is estimated as:

$$TW_{fcf} = \frac{\lambda_{fcf} * E[TS_{fcf}^2]}{2(1-\rho_{fcf})}$$

2.4 Disk array subsystem

The most complicated service component in this SAN configuration is the high-end storage system. There are three different service nodes within the storage system: DACC, FC-AL network, and disks.

In an I/O read request, the disk array cache is searched for the requested data. If the data is found in the cache (known as a cache hit), the DACC will respond to the host I/O request directly without any disk operation. If a cache miss occurs, the DACC will then issue the I/O read command to the related hard disks through the FC-AL connection. Read request issued by the DACC depends on the I/O size and RAID parity distribution algorithm. If the I/O request size is smaller than one striping size, DACC issues one read request to one hard disk. If I/O size is larger than one striping size but smaller than one trunk size, then DACC issues multiple read requests to different disks. If the I/O size equals to or is larger than one trunk size, DACC issues N_d read request to all disk drive (N_d) of the parity group. Therefore, read request rate issued by the DACC is affected by the cache hit rate, I/O size and RAID parity distribution policy. The RAID parity placement algorithm used in this paper is the right asymmetric or left symmetric RAID5[10].

When a write command is issued to the disk array, the DACC will first check the cache to see if the data blocks to be written are already stored in the cache (from previous read or write commands). If they are, the respective cache segments are cleared. In the event that write caching is enabled, then the DACC returns status information to host, and data in the cache can be de-staged to disks according to the optimization algorithm of the disk system. If write caching is disabled, the DACC will not return host information until data is written to the disks.

When DACC de-stages the data from disk array cache to disks, it also generates parity information and issues parity request to disks to guarantee the disk array reliability. Obviously, the write request issued by DACC (including de-staging request and parity request) depends on the write cache hit rate, de-stage size, striping size, disk array size, and parity placement schedule. For small random write operation in a normal RAID5 read-modify-write rule, the DACC will issue additional read requests.

When an I/O command issued by DACC involves multiple disks operation (one disk I/O request for each disk), then the response time of the DACC's I/O command is defined as the time period between the first and the last DACC command of the I/O request family to leave all service nodes. Therefore, the I/O command response time and service time consist of both the time of the single service node and multiple disk synchronization. A fork/join model is used to analyze the performance of the disk array [7].

The response time in a Fork/Join model for *k* number of disk drives is given by:

$$TR_k(\lambda) = TS_k(0) + \frac{\delta * \mu_{disk}^2 * \rho_{disk} * E[TS_{disk}^2]}{\mu_{disk} - \lambda_{disk}}$$

Where,

$$\delta = 0.5 * (H_k + (4 * V_k - 3H_k - 1) * \beta + 2(1 + H_k - 2V_k)\beta^2)$$

$$H_k = \sum_{i=1}^{k} \frac{1}{i}$$

$$V_k = \sum_{i=1}^{k} C_k^i (-1)^{i-1} \sum_{j=1}^{i} C_i^j \frac{(j-1)!}{i^{j+1}}$$

$$\beta = \frac{E[TS_{disk}]^2}{E[TS_{disk}^2]}$$

$$TS_k(0) = E[TS_{disk}] + D[TS_{disk}]\sqrt{2\log k}$$

Where, λ_{disk} is the I/O request rate of the single disk drive issued by DACC.

For read operation:

$$\lambda_{read} = \alpha * (1 - hitread_{dacc}) * readratio * \lambda_{dacc}$$

Where,

$$\alpha = \begin{cases} 1 & for \quad IOsize \le stripingsize \\ Ceil(IOsize / stripingsize) & for \quad stripingsize < IOsize < trunksize \\ N_{disk} & for \quad IOsize \ge trunksize \end{cases}$$

For write operation

$$\lambda_{write} = \quad * (1 - readratio) * (1 - hitwrite_{dacc}) * \lambda_{dacc} * N_d / (N_d - 1)$$

$$\partial = \begin{cases} 1 & for \quad destagesize \le stripingsize \\ Ceil(destagesize / stripingsize) & for \quad stripingsize < destagesize < trunksize \\ N_{disk} & for \quad destagesize \ge trunksize \end{cases}$$

Then total I/O request for FC-AL network is given by,

$$\lambda_{fcal} = readratio * \lambda_{read} + (1 - readratio) * \lambda_{write}$$

Average I/O request rate for single hard disk is

$$\lambda_{disk} = \lambda_{fcal} / N_d$$

2.4 FC-AL connection

Most high-end storage systems use Fibre Channel storage protocol, which is configured in an Arbitrated Loop (AL) topology. FC-AL protocol permits each L_Port to arbitrate access to the Loop. Priority is assigned to each participating L_Port based on the Arbitrated Loop Physical Address (AL_PA). This could lead to situations where the L_Ports with lower priority will not be able to gain access to the Loop. Hence, the "access fairness" algorithm is required to set up an access window in which all L_Ports are given an opportunity to arbitrate and win access to the Loop. After all L_Ports have received opportunities to access the Loop, a new access window is started.

In the disk array configuration herein, there is only a single initiator (DACC) and many target devices (disk drives) connected to the Loop. If all Ports on the Loop including the DACC are accorded the "access fairness" algorithm, the DACC may not be able to obtain sufficient Loop bandwidth to achieve a high level of parallelism among disk drives and optimize overall performance. For example, in the situation where the DACC transmits read commands to all disk drives, the disk drives may require some time to prepare the read data before transmitting to the DACC. Once a disk drive acquires the Loop priority, it remains inactive unless it is given another command. If the DACC follows the fairness algorithm, it will have to wait for all disk drives with pending read data before being able to access the Loop to send new commands. To reduce this disk drive inactivity, the DACC need to "unfairly" acquire the Loop to issue new commands. The disk drive is able to receive multiple commands in its command queue.

Four different FC-AL schedules are analyzed for the command waiting time described above. The FC-AL schedules considered include the Fairness Access Algorithm (FAA) whereby all command and data requests have equal priority, the Read Command First (RCF) Algorithm which dictates higher priority to read command and normal priority to all other requests, the FL-Port First (FLF) Algorithm that gives higher priority to requests issued by the FL_Port and all other requests having equal priority, and finally the Command First (CF) Algorithm that provides higher priority to all command requests and the others have normal priority.

In the SAN performance analysis, the CF schedule is used in the FC-AL network model. The I/O response time serviced by FC-AL network is estimated as:

$$TR_{fcal} = TW_{cmd} + TW_{data} + TS_{fcal}$$

Where, TW$_{cmd}$ and TW$_{data}$ are command and data request waiting time.

$$TW_{cmd} = \frac{\lambda_{fcal} * E[TS_{fcal}^2]}{2(1-\lambda_{fcal} * E[TS_{cmd}])}$$

$$TW_{data} = \frac{\lambda_{fcal} * E[TS_{fcal}^2]}{2(1-\lambda_{fcal} * E[TS_{cmd}])(1-\lambda_{fcal} * E[TS_{fcal}])}$$

2.5 Disk unit

Current disk drives have read/write cache to improve disk performance. This disk cache together with the HDA, are the two service nodes used to model the disk unit herein. The function of this disk cache is similar to that of the disk array cache. When the read/write cache is enabled, data requested by the initiator is retrieved from the buffer before any disk access is initiated. If the data requested is already in the cache, the drive will respond to the initiator immediately. The pre-fetch feature of a disk allows data in contiguous logical blocks to be retrieved beyond that which was requested by a read command and stored in the buffer. Subsequent read commands can then immediately transfer the data from the buffer to the host. Pre-fetch size affects cache hit rate and the performance of disk.

For write operation with write caching enabled, the disk drive will return a positive status acknowledgement after the data has been transferred into the cache, but before the data is written to the medium. After that, data in the cache will be de-staged to disk media to optimize the system performance. If an error occurs during de-staging but the positive status has already been returned, a deferred error will be generated. To avoid this, write caching function can be disabled. However, write cache is always enabled in the model described in this paper.

From the above description, if disk read/write cache is enabled, the request rate and request size of the disk are different from the original disk I/O request. For example, in a sequential disk I/O of 4kb size, the de-stage size between disk cache and disk media can be set equal to the track size. Then, there will be no overhead for seek operation and rotational delay for disk media access because pre-fetch and write cache is enabled. The HDA service time will consists of data transfer time, head switch time, and cylinder switch time. But if the I/O request is randomly distributed in the disk media, the seek time and rotational delay have to be calculated as the basic HDA service time.

The response time for hard disk can be estimated as:

$$TR_{disk} = TR_{dcc} + P_{hda} * TR_{hda}$$

$$TR_{dcc} = E[TS_{dcc}] + \frac{\lambda_{dcc} * E[TS_{dcc}^2]}{2(1-\lambda_{dcc} * E[TS_{dcc}])}$$

$$TR_{hda} = E[TS_{hda}] + \frac{\lambda_{hda} * E[TS_{hda}^2]}{2(1-\lambda_{hda} * E[TS_{hda}])}$$

Where, TR_{dcc} and TR_{hda} are response time for DCC and HDA centers respectively, P_{hda} is probability of the I/O request to access disk media.

$$\lambda_{hda} = \lambda_{dcc} * (1 - hitrate\_dcc + hitrate\_dcc * blocksize / disk\_destagesize)$$

$$TS_{hda} = (1 - hitrate\_dcc) * (t_{seek} + t_{la}) + t_{dt} + p_{hs} * t_{hs} + p_{cs} * t_{cs}$$

Where, $p_{hs} = destage\_size / track\_size \quad p_{cs} = destage\_size / cylinder\_size$.

3. Results and analysis

3.1 SAN system throughput

The I/O performance metrics used are Throughput (MB/s) for bandwidth, IOPs for transaction processing, and Response Time for detail analysis of the server nodes. If there is an infinite queue depth, the throughput would simply be equal to *1/E[TS]*. In actual case, system performance is limited by either the queue depth or response time.

Figure 2(a) shows the relation between utilization and system queue depth and Figure 2(b) shows variation of queue waiting time with queue depth for three kinds of the different queue models. When the queue depth reaches to 8 to 16 orders, most nodes utilization will reach approximately 95%. Further increase to the queue depth only provides slight improvement to the utilization, but a tremendous increase in response time. In this paper, the system performance (*IOPs* and throughput) is defined as the actual system request rate (*IOPs*) and request data throughput (*IOPS*I/O size*), when the maximum utilization index of a service node reaches 0.94 (94%).

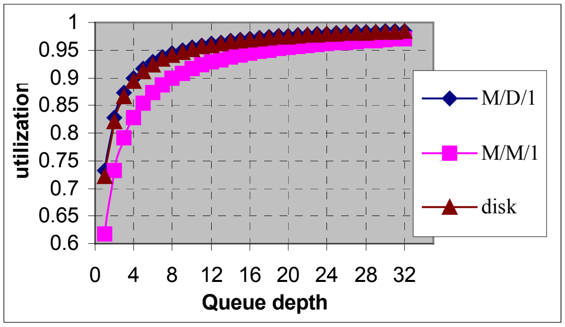

Figure 2(a). Utilization varies with queue depth for various SAN service model.

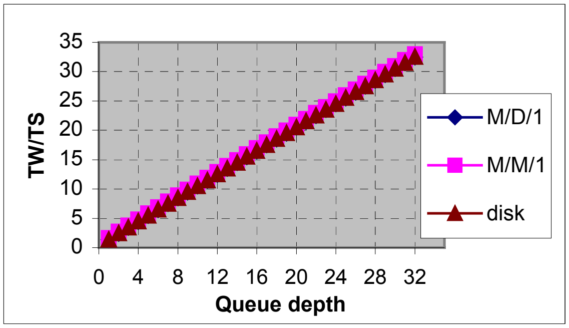

Figure 2(b). Variation of queue waiting time with queue depth for various SAN service model.

3.2 SAN performance

Firstly, four typical cases are analyzed; sequential big I/O, random big I/O, sequential small I/O and random small I/O. The basic input parameters are shown in Table 2. Figure 3 shows that the system response time and its distribution vary with the system actual throughput for sequential read operation with 1Mbytes I/O size. The correspondent utilizations of each service node are shown in Figure 4. The basic configuration consists

Table 2: Parameters used in calculation

Host:
PCI bus: 64bits, 33MHz;
Command overhead: 0.2ms (including OS, device driver and controller overhead)
FC network:
1.0625Gb/s;
DACC:
Striping size 32KB;
Disk number: 5/15/25;
Trunk size: 32*(disk number-1)
PCI bus: 64bits, 66MHz;
Read overhead: 0.16ms
Write overhead: 0.18ms

Hard disk drive:
Controller overhead: 0.2ms (from receive of the FCP command to the FCP response);
RPM: 10025;
Track per surface: 14100
Track size: 256KB;
head switch time: 0.8ms
cylinder switch overhead: 1.2msec
seek time:
read: 0.6+0.0876*sqrt(x);
write: 0.9+0.0910*sqrt(x);
x is seek distance.

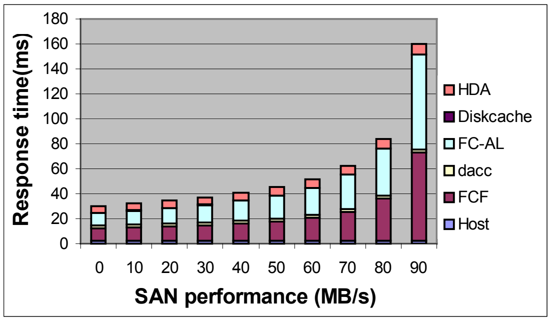

Figure 3. System response time varies with system throughput for sequential 1MB I/O size.

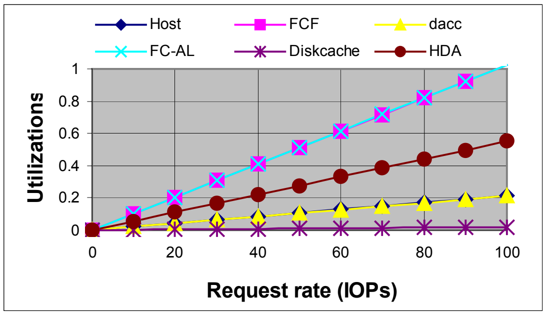

Figure 4. Utilizations of service nodes vary with system request rate for sequential 1Mb I/O size.

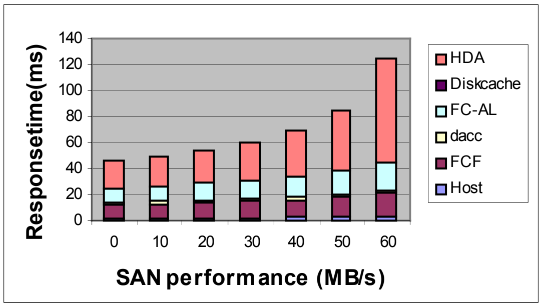

Figure 5 System response time varies with system throughput for random 1MB I/O size.

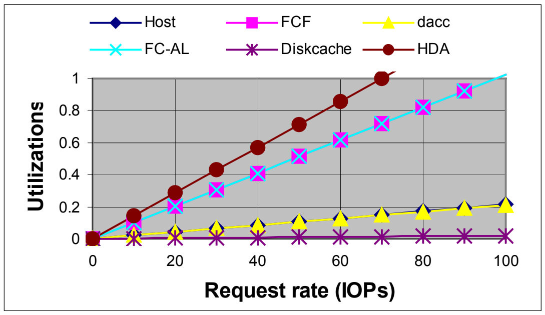

Figure 6 Utilizations of service nodes vary with system request rate for random 1MB I/O size.

of one host, one FC fabric switch and one RAID array system made of one controller and five disks. From Figure 3, it is observed that the response time in the FC network is occupied 60% to 90% depending on the system actual data flow. The maximum throughput for current configuration is 94MB/s. Figure 4 shows that when the system throughput reaches 94MB/s, the hard disk utilization only reaches 50% and utilizations of other nodes are even lower. In other words, the FC network has become the I/O performance bottleneck of the SAN.

The response time and utilization of SAN service nodes for random 1MB I/O size are shown in Figure 5 and 6. The figures demonstrate clearly that the HDA response time is the largest portion (40%~60%) of the entire response time distribution. Figure 6 also shows that the system performance is limited by the hard disks and the maximum performance of the SAN in this configuration is 66MB/s. In order to identify the system bottleneck and effects of number of disks, we analyze the same SAN configuration with 15 disks and 25 disks. The maximum system throughput is 79MB/s and 82MB/s respectively. The system performance improved slightly with increasing disk numbers, but it is still limited by the hard disk performance.

To evaluate the effects of the I/O size on the performance of SAN, small sequential and random I/O (4kb size) are analyzed. Figure 7 and 8 show that, for sequential I/O, the I/O response time and utilization of service nodes vary according to actual system performance. From the response time distribution, it is observed that the system performance is limited by the HBA overhead. The maximum system I/O performance for

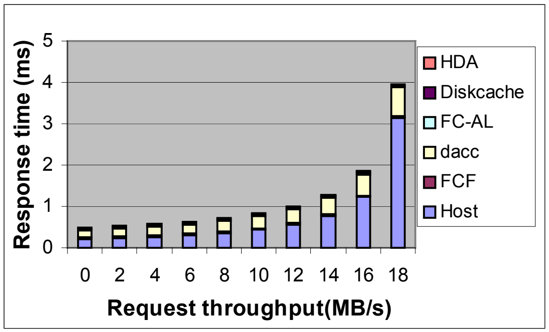

Figure 7. System response time varies with the throughput for sequential 4KB I/O (single user).

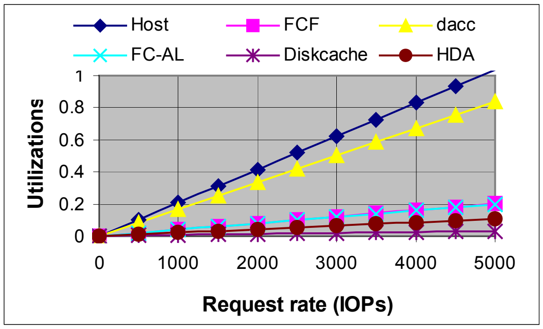

Figure 8. Utilizations vary with system request rate for sequential 4KB I/O (single user).

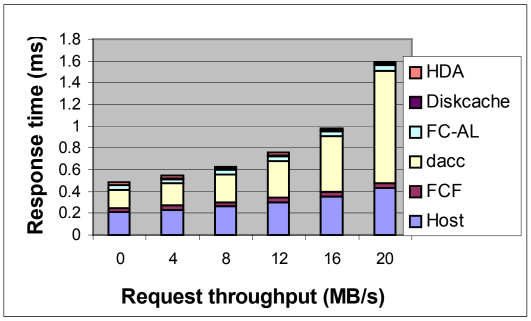

Figure 9. System response time varies with the throughput for sequential 4KB I/O (2 users).

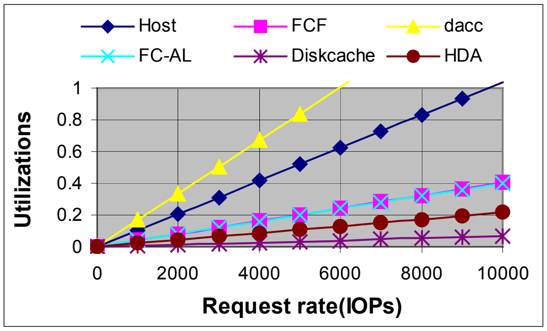

Figure 10. Utilizations varies with the system request rate for sequential 4KB I/O (2 users).

single user configuration is 4500 IOPs. In order to identify the maximum performance of the storage system, we evaluate a configuration with 2 hosts. Figure 9 and 10 show that the response time and utilization vary according to I/O performance of the SAN configuration with 2 similar users. The main part of the response time distribution is contributed by the DACC. In other words, the DACC is the system performance bottleneck and the maximum performance is 5600 IOPs.

For the case of small random I/O, Figure 11 & 12 show that the response time and utilization also vary with system I/O performance. It is assumed that there is no tagged queue for I/O command and optimization of the seek operation for multiple I/O access. The maximum performance of the SAN is only with 420 IOPs. This is due to low cache hit rate in the disk array cache, causing each I/O request to rely on disk media access. The utilization of hard disk is far larger than that of other nodes. The performance is limited by the disk drive mechanical delay. Further evaluations are conducted by increasing the to 15 disk drives in the array. Figure 13 and 14 show the response time and utilization of service nodes variation with the system I/O performance of the 15 drives configuration. The figures shows that the mechanical delay of the hard disk is still the main contributor of the system response time. The system performance has increased to 1260 IOPs, which is approximately 3 times of the previous performance. When the disk array is futher increased to 25, calculation shows that the system performance will increase to 2100 IOPs. This means that adding more disk drives to an array is an effective method of improving the random I/O performance.

3.3 Disk drive performance

From the above analyses, the effect of the disk cache is not always the bottleneck of the

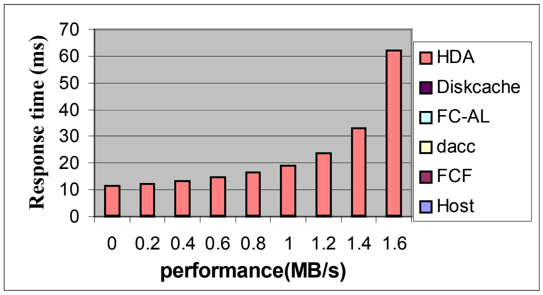

Figure 11 System response time varies with the throughput for random 4KB I/O (5 HDDs).

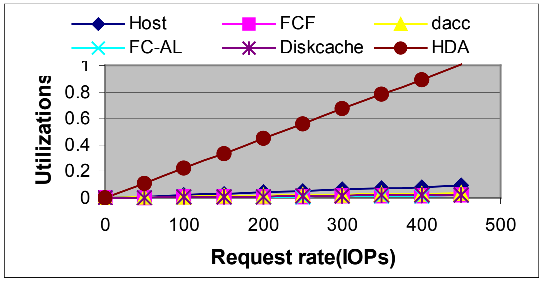

Figure 12 Utilizations vary with the system request rate for random 4KB I/O (5 HDDs).

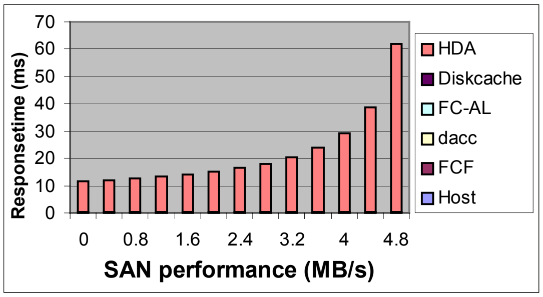

Figure 13 System response time varies with the throughput for random 4KB I/O (15 HDDs).

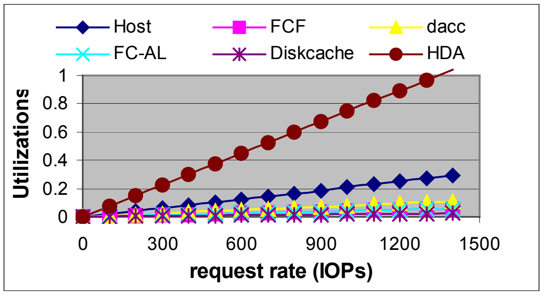

Figure 14 Utilizations vary with the system request rate for random 4KB I/O (15 HDDs).

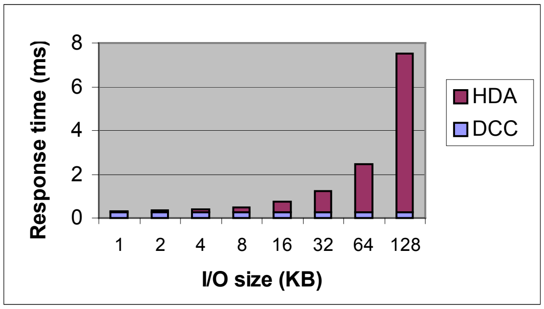

Figure 15 Response time varies with I/O size when the request rate is 200IOPs for hard disk.

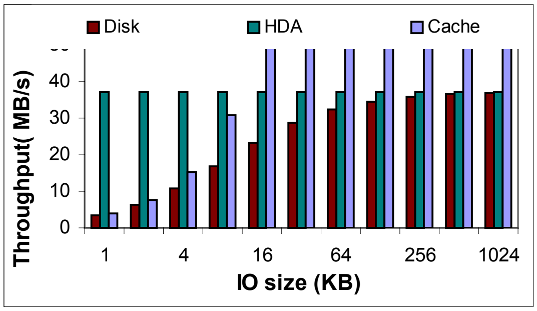

Figure 16 Bandwidths of the disk unit, DCC and HDA vary with IO size for single hard disk.

SAN system performance. A detail analysis of the disk performance is conducted. Figure 15 shows that response time varies with I/O size when the I/O performance is 200 IOPs in sequential disk I/O. The overhead of the HDA becomes the main part of system response time as the I/O size increases to 16KB. Figure 16 shows the bandwidth of the disk unit, DCC and HDA for sequential I/O. It is observed that the DCC node is the limitation when the sequential I/O size is smaller than 8KB. When the disk I/O size is larger than 16 KB, the performance of the disk is not limited by the DCC overhead any more. In our above SAN performance analysis, the disk array striping size is selected as 32 KB. Therefore, the SAN performance is not limited by the overhead from the disk cache.

3.4 Disk array performance

For large I/O in a disk array, the I/O operation involves multiple hard disks. As mentioned, the response time of such I/O request includes both the disk drive response time and synchronization time. To analyze the effects of Fork/join model on the SAN I/O performance, we define 'ratio' for access time and waiting time as the ratio of the Fork/Join access time and Fork/Join queue waiting time divided by the basic access time and queue waiting time for single drive. Figure 17 show the access time and queue waiting time vary with the disk drive number. For current hard disk drive, the access time ratio for disk array with less than 30 disk drives is between 1 and 1.45. The waiting time

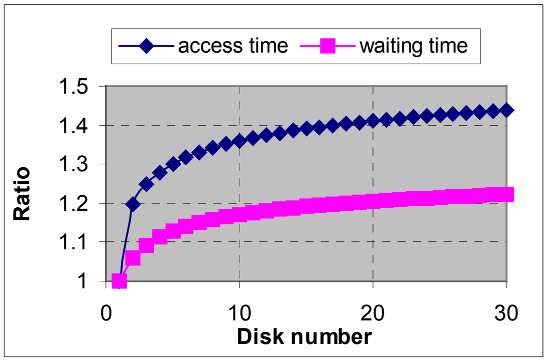

Figure 17 Fork/Join access time and queue waiting time ratio vary with disk numbers.

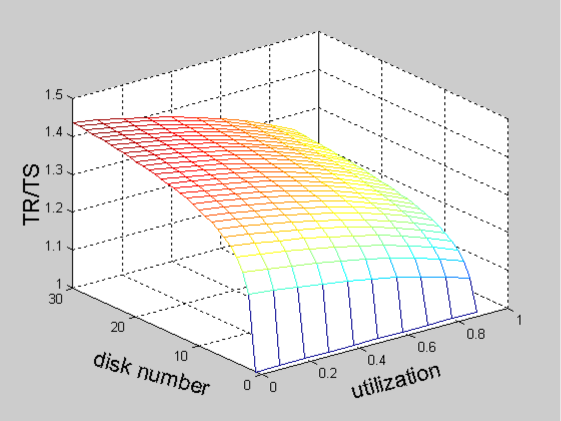

Figure 18 Response time ratio varies with disk numbers and system utilization.

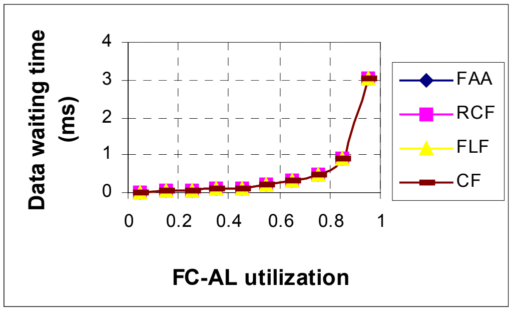

Figure 19 Data average waiting time varies with FC-AL utilization.

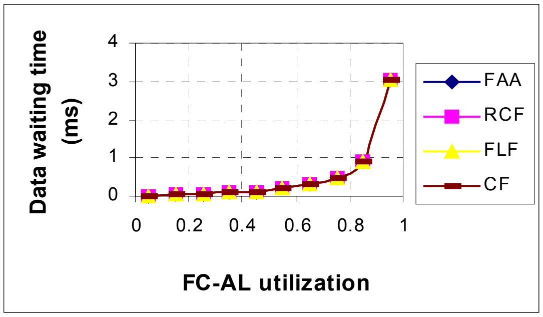

Figure 20 command average waiting time varies with FC-AL utilization.

ratio is between 1 and 1.2. Figure 18 shows that the response time ratio varies with disk number and system utilization. From figure 18, the response time loss induced by the multiple disk synchronization lies between the access time ratio and waiting time ration. It can reach 20% for system with high utilization.

3.5 FC-AL performance analysis

Comparison between the different FC-AL schedule policies: fairness access algorithm (FAA), read command first (RCF), FL-Port First (FLF), and Command First (CF) has been conducted. Figure 19 and 20 show that the data average waiting time and command waiting time vary with system utilization respectively. The read ratio is assumed to 0.5. There is little difference between the data average waiting time. But when FC-AL utilization is high to 0.75~0.95, there is obviously difference between the command average waiting time. The command waiting time is a very important factor that affects the performance of the disk array with FC-AL topology. The command waiting time of the CF schedule is of the minimum value.

3.6 Comparison between theoretic and testing results

An experimental testing is conducted to compare with theoretical results. The tests are based on the multiple Pentium 850 hosts with 64bits and 66MHz PCI bus, HBA with Qlogic QLA 2200A, 1G FC switch with Brocade Silkwarm 2400, and a self-developed

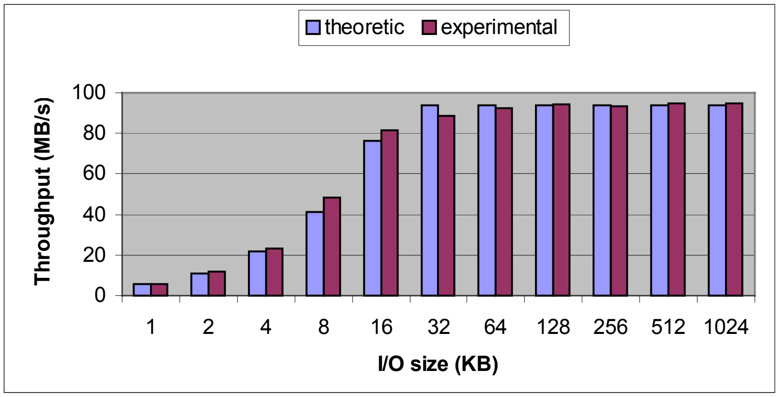

Figure 21. The comparison of the theoretic and experimental result

virtual FC Disk. IOMeter is used as the benchmark tool.

Figure 21 shows the theoretic and testing throughputs vary with I/O size for sequential read. Both of the testing and theoretic results show that the FC network is the system bottleneck for big I/O size (>32KB), and the storage system controller overhead is the system limitation for small I/O size (<16KB). The analytical results are in agreement with experimental results.

4. Conclusion

We have developed a theoretical model for analyzing the overall performance of the storage area network as well as the disk array system. The system bottleneck lies on the different nodes for different I/O request and application. For random I/O, the most possible bottleneck is hard disk drives. For large I/O, the FC network bandwidth is the most possible bottleneck. For small sequential I/O, the most possible bottleneck is the disk array processing and controller. The performance of the hard disk drives with a Fork/Join model are analyzed also. The results show that cache overhead is not the bottleneck of the whole SAN network, if the striping size of the disk array is bigger than 16KB. The performance loss caused by multiple drive synchronization can be as much as 20%. The schedules for the FC-AL protocol to transfer commands and data were analyzed, and the Command First (CF) schedule is recommended. The theoretical results are compared with the test results, and found to be in agreement.

References

[1] E. Shriver, B. K. Hillyer, and A. Silberschatz, “Performance analysis of storage system”, Performance Evaluation, LNCS, pp33-50, 2000.

[2] C. Ruemmler and J. Wilkes, “An introduction to disk drive modeling”, IEEE computer, vol.27, pp17-28, Mar. 1994.

[3] C.S. Lee, and T.M. Parng, “Performance modeling and evaluation of a two-dimensional disk array system”, Journal of Parallel and Distribution Computing, 38, 16-27, 1996.

[4] T. Ruwart, “Disk Subsystem performance evaluation: from disk drivers to storage Area Networks”, proceeding, 9th NASA Goddard Conference on Mass Storage Systems and echnologies, April 2001.

[5] J. R. Heath, and P. Yakutis, “High-speed storage area networks using a Fibre Channel arbitrated loop interconnect”, IEEE Networks, March/April 2000.

[6] Michael Smith, “Benefits of a large HBA data buffer”, INFOSTOR, July 2001.

[7] S. Varma, and A. M. Makowski, “Interpolation approximations for Symmetric Fork-Join Queues”, Performance Evaluation Vol.20, p245-265, 1994.

[8] ANSI, “Fibre Channel Arbitration Loop, Draft proposed –ANSI Standard X3T11, Rev. 4.5, 1995.

[9] A.O. Allen, “probability, statistics, and queueing theory with computer science applications”, second Edition, Academic Press, 1990.

[10] E.K. Lee and R.H. Ratz, “Performance consequences of parity placement in disk array”, IEEE Trans. Computer, Vol. 42, pp651-664, June 1993.