On the Effect of Link Failures in Fibre Channel Storage Area Networks

Xavier Molero, Federico Silla, Vicente Santonja and José Duato

Departament d'Informàtica de Sistemes i Computadors
Universitat Politècnica de València
Camí de Vera, 14. 46022 València (Spain)
e-mail: {jmolero,visan}@disca.upv.es, {fsilla,jduato}@gap.upv.es

Abstract

The fast growth of data intensive applications has caused a change in the traditional storage model. The server-to-disk approach is being replaced by storage area networks (SANs), which enable storage to be externalized from servers, thus allowing storage devices to be shared among multiple servers. The prominent technology for implementing SANs is Fibre Channel, due to its suitability for storage networking.

Although the probability of a link failure for individual links in a SAN is very low, this probability dramatically increases as the network size becomes larger. Moreover, there are external factors, such as accidental link disconnections, that also can affect the overall SAN reliability. Until the faulty element is replaced, the SAN is functioning in a degraded mode.

In this paper we analyze by simulation the performance degradation of Fibre Channel storage area networks when failures in links occur, quantifying how much the global SAN performance is reduced during the time the system remains in the degraded state. We perform this analysis by using both synthetic and real I/O traffic. Simulation results show that performance degradation mainly depends on the routing algorithm and the switch architecture used.

1 Introduction to Storage Area Networks

The rapid growth of data intensive applications, such as system simulation, modeling, Internet and intranet browsing, multimedia, transaction processing, e-business, and data warehousing and mining are continously driving the demand for more data storage capacity [15]. Moreover, at the same time that network bandwidth is constantly increasing, the ever more powerful network clients continue to overburden traditional file servers. This situation, as well as the growth of networking in organizations, has caused that large companies may have hundreds of servers distributed over tens of sites, storing terabytes of data that contain vital information. The dependence of these organizations on the networked data is higher than ever before.

On the other hand, the reliance on the access to data is emphasizing the limitations of traditional server-storage solutions. In these environments, each server has its own private connection to storage devices, usually based on SCSI buses [15, 19]. However, this approach is now facing several problems. The first one is related to the availability of data. This is an important issue, because as different studies have shown, data unavailability could be very expensive for a company. If the sever becomes unavailable, there is no way to access the data in its storage devices, since the server controls all accesses to the data. Performance is another issue. SCSI performance is limited to 20 or 40 MB/s with SCSI-II, and 40 or 80 MB/s with Ultra SCSI. This bandwidth is shared among all the devices attached to the bus. Additionally, performance is limited by the server's capabilities and loading. Also, configuration flexibility is limited by the SCSI interface, a standard that has been around for about 15 years. SCSI technology suffers from a number of limitations, such as the maximum cabling length, which is 25 meters, or the number of devices per cable, which is limited to 16 devices. This latter constraint causes that large configurations require several buses.

Additionally, since stored information represents the most valuable resource in most companies, it is essential to protect it, being necessary to perform efficient, consistent, and regular backups. However, with the traditional storage approach, system managers confront a problem: the amount of data to be backed-up is exploding, and at the same time, the demands for e-mail, e-commerce, and web-hosting require continuous uninterrupted operation (24 hours a day, seven days a week). Thus, the amount of time available for

1087-4089/00 $10.00 © 2000 IEEE

backup is decreasing. These demands cause performance bottlenecks during the time allocated for backups. Moreover, in the case for multiple servers, massively moving files during backup between servers and storage devices saturates network bandwidth. During these operations, users may experience unacceptable long response times.

An emerging alternative to the traditional server-to-disk approach is based on the concept of storage area network (SAN). A SAN is a high-speed network, similar to a LAN, that establishes an indirect connection between storage elements and servers (see Figure 1). A SAN can use interconnect technologies similar to the ones used in LANs, namely hubs and switches. In the same way that LANs allow the clients easy access to many servers, SANs provide easy access for multiple servers to multiple storage devices.

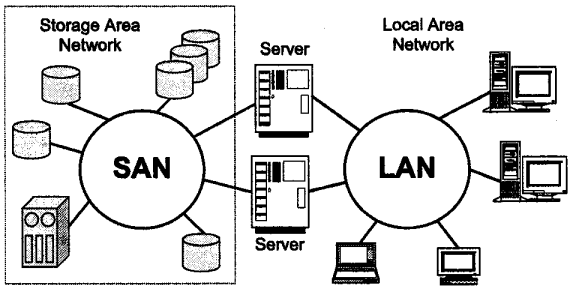


Figure 1. A typical SAN-LAN environment

SANs enable storage to be externalized from the server and by doing so, allow storage to be shared among multiple servers without impacting the performance of the primary communications network, whose perfor- mance improves because it is freed from the high overhead associated with file access retrieval, storage, and data backing-up functions. Furthermore, data access and availability are enhanced because file read/write, back-up/restore, data migration, and data and device sharing are more effectively handled by a network that can be optimized for the demands of storage operations (e.g., high throughput and large packet data transfers). Moreover, in a SAN, stored data is accessible by way of alternative data paths, thus providing fault-tolerant operation.

With respect to backing-up data, the deployment of a SAN is a cost-effective way to move traffic generated by backups from the primary LAN to a storage subnet, thus freeing LAN bandwidth, which could be used to run applications more efficiently and avoid the down times required for periodic backups. The SAN architecture facilitates the movement of data directly from one storage device to another (e.g., from disk to disk, disk to tape, or tape to disk). In this way, the data will bypass both the server and the LAN. The role of server during backup is reduced to coordinate the whole operation.

The SAN architecture also allows storage devices to be physically at different sites. In these storage area networks spanning one or several buildings, it is possible that some links may suffer accidental disconnections, or they can be affected by electromagnetic interferences (EMI). At the moment a link fails, the topology of the network changes and a process of reconfiguration starts in order to compute the routing tables for the new topology. During this process, a distributed reconfiguration algorithm is usually used in order to make the network operational as soon as possible [17, 1, 3]. However, after the reconfiguration phase, the network will have a lower aggregate bandwidth than before the failure. Also, after the reconfiguration, connectivity between servers and disks will be degraded.

Managing failures has been analyzed in the context of direct networks with regular topologies [2, 4, 7, 9] and also in the context of networks of workstations (NOW) with irregular topologies [14]. In the case for regular networks, a fault-tolerant routing algorithm is needed in order to bypass the faulty region. These algorithms are usually topology dependent. In the case for networks of workstations, where generic routing algorithms must be used due to the topology irregularity, managing failures may be faced in a more general way. In these networks, every time a new workstation is connected or disconnected to/from the network, it is necessary to run a reconfiguration process, in order to update routing tables. Also, every time a switch is attached/unattached to the network, routing tables must be updated. Therefore, managing failures in irregular networks may be seen as another instance of the general reconfiguration process. In the case for storage area networks, they usually support irregular topologies, as networks of workstations. In fact, routing algorithms used in SANs may be the same as the ones used in NOWs. Therefore, failures may be managed in the same general way as they are managed in NOWs. However, besides the similarities in the way both types of networks deal with failures, the context is very different, mainly due to networks size and load characteristics.

In this paper we will focus on analyzing the impact on the performance of SANs when functioning in a degraded mode once the network has been reconfigured after the occurrence of a link failure. A study of how much network performance is degraded would be useful to know how much I/O response times are affected by link failures. Among other parameters, a key issue that highly affects network performance is the routing algorithm used by messages to reach the destination node.

As mentioned before, a faulty link modifies network topology. Thus, it is important to know how much partial modifications in the topology will affect the performance of the routing algorithm used by I/O operations. In this way, we have evaluated the sensitivity to failures of two different routing algorithms: the partially adaptive up*/down*

routing algorithm [17], and the minimal adaptive routing scheme [18]. We have also analyzed the influence of the internal switch architecture on network performance once it is functioning in degraded mode. In this sense, we have considered two different switch architectures for Fibre Channel switches [5, 13].

The remainder of this paper is organized as follows. Section 2 presents a detailed description of the different parameters of the storage area networks we have considered, such as the switch architecture, the network and link failure models, the communication pattern between servers and disks, and the I/O load characteristics. Section 3 presents the evaluation methodology and an analysis of the performance results, where both synthetic and real I/O traffic have been used. Finally, Section 4 summarizes the conclusions from our study.

2 System Model and Evaluation Metodology

This section describes the different aspects of the SAN model used in order to obtain realistic values for the SAN performance degradation under the presence of link failures.

2.1 Fibre Channel Switch Architectures

The switch architecture proposed in [5] for Fibre Channel is shown in Figure 3. Each input port has an associated pool of buffers to store incoming messages (called *frames* in Fibre Channel terminology). When a message arrives at a port, it starts being read into one of the buffers attached to that port. The routing unit iteratively polls input ports, in a round-robin scheduling policy, for new messages that need to be routed. This unit can start routing a message as soon as the header information has arrived. If it finds new messages in a port, the router selects the first one and reads the header information, which contains the destination address. Then it inserts a link to the message into a scheduler table, organized as FIFO queues, one for each output port. The scheduler table is used to control the access to the internal crossbar. This access must take into account the following three rules. First, the crossbar can transfer at most one message from each input port at any given time. Second, it can transfer at most one message to each output port at any given time. Finally, a message can be transferred to the selected output port only when its link in the scheduler table is at the top of the queue for that output port.

Fibre Channel uses distributed routing, leaving routing decisions to each of the switches in the path from source to destination. Moreover, Fibre Channel uses the virtual cut-through switching mechanism [11]. We have considered that there are eight buffers per input channel. Also, since data length of I/O transfers usually is a few multiples of

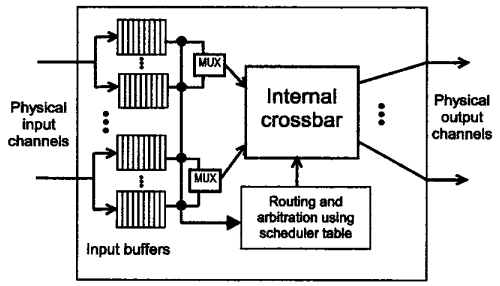


Figure 2. Fibre Channel switch architecture I

1024 bytes. We have assumed that each input buffer can store 2048-byte messages[1].

Fibre Channel switches have duplex serial links at 100 MB/s. Link length may range from a few meters until several kilometers when optical fiber is used. The flow control mechanism used in Fibre Channel to transfer data from one switch to another is based on credits. A credit represents the switch availability to accept an additional message. The credit scheme is based on the number of input buffers per physical channel. Each switch has a counter per output channel that is initially set to the number of input buffers per physical channel. In this way, the sending switch knows at every moment the buffer availability at the receiving switch. Each time the sending switch forwards a message, it decrements the corresponding counter. When the counter reaches the zero count, no message can be sent through that output channel. On the other hand, every time an input buffer is freed, the receiving switch sends back a "credit" message so that the sending switch increments the corresponding counter.

Additionally, we have considered an improved Fibre Channel switch architecture proposed in [13]. In this architecture, shown in Figure 3, link bandwidth has been increased, according to recent proposals for gigabit networks [1], upto 160 MB/s. Moreover, the routing unit has been improved removing the scheduler table. Therefore, if the requested output link is busy, the incoming message remains in the input buffer until it it successfully routed. Also, the internal crossbar used is a non-multiplexed one, with as many input ports as input buffers, thus increasing the crossbar connectivity and also its implementation cost.

Switch and link parameters used in our study have been taken from actual high-speed interconnects. Thus, the time needed to route and forward the first byte of a message is 150 ns. Following bytes take 6.25 ns to traverse the switch, and 50 ns to travel along a physical link, which presents a delay of 1.5 ns/ft (we have assumed 10 meter cables). Because our simulator uses clock cycles instead of ns, we will

[1]The Fibre Channel standard specifies that the data field for messages may range from 0 to 2112 bytes.

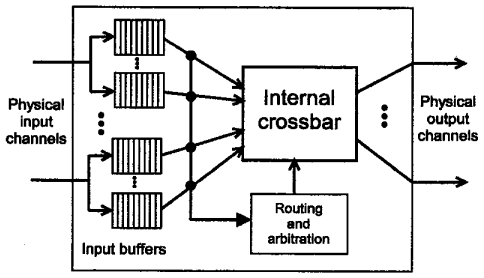


Figure 3. Fibre Channel switch architecture II

assume the equivalence that a clock cycle is 6.25 ns. Therefore, the time needed to compute the routing algorithm will be 23 clock cycles. Also, it will take one clock cycle to transmit a byte across the internal crossbar. Link propagation delay will be 8 cycles.

On the other hand, depending on the switch architecture used, link bandwidth will be 160 MB/s or 100 MB/s. Therefore, data will be injected into the physical channel, which is pipelined, at different maximum rates. In the case for 160 MB/s links, data will be injected into the channel at a rate of one byte per cycle, while in the case for the 100 MB/s link, a new byte will be injected into the physical channel every 1.6 clock cycles.

2.2 Network and Link Failure Models

The network is composed of a set of switches. Network topology is completely irregular and has been generated randomly. However, for the sake of simplicity, we imposed three restrictions to the topologies that can be generated. First, we assumed that there are exactly one server and four disks connected to each switch. Second, two neighboring switches are connected by a single link. Finally, all the switches in the network have the same size. We assumed 8-port switches, thus leaving three ports available to connect to other switches. Regarding network size, we have considered networks composed of 8 and 16 switches (8 servers and 32 disks, and 16 servers and 64 disks, respectively). These network sizes may fit the needs of small and medium-sized organizations.

On the other hand, the location of link failures is chosen in a random way. The only restriction is that the resulting topology after the link failure must be connected.

2.3 Communication Between Servers and Disks

In a SAN environment, servers initiate the communication with disks, and thus disks actions are determined by the requests previously sent by servers. Requests are defined by a set of parameters. Some of them identify the access characteristics, such as disk, type (read or write), starting location, and size. For I/O read operations, the server first sends a request message to the selected disk indicating the number of sectors to be read, and the initial sector address in the disk. Once the disk has processed the request, it accesses the information, and sends back the data to the server through the storage network. Regarding I/O write operations, servers start them by sending the data and the initial sector address to the disk. Once the data is stored, the disk returns an acknowledgment message to the server. The I/O write operation lasts since the server initiates it until the acknowledgment is received.

The acknowledgment for writes and the request for reads are both short messages of a small amount of bytes (typically, 16 or 32 bytes). These messages are called control messages. The size of data messages depends on file system and disk interfaces, but usually may be a few KB long. As a result, the load supported by the storage interconnection network becomes bimodal, being composed of long and short messages in an equal percentage. On the other hand, in this work we have consider only delivery class 3 service (connectionless), in which acknowledgment messages are not generated for each received data or control message.

2.4 I/O Load Characteristics

Modeling I/O load is a difficult task. Ruemmler and Wilkes [16] presented an exhaustive analysis of a set of I/O traces, providing detailed information about disk accesses in several systems and concluding that many results are significantly different from those described in the literature. Moreover, Ganger [8] showed that commonly used simplifying assumptions about storage loads are incorrect. In particular, by analyzing arrival patterns, he concluded that disk arrival processes do no match those generated by Poisson processes. Finally, Gómez and Santonja [10] have presented evidences confirming that I/O arrival patterns are consistent with self-similarity, meaning that real traffic is bursty and that the bursts exist over many time scales.

Taking into account these considerations, we have divided our study into two stages. In the first one, in order to easily understand the effect of link failures on SAN performance, we have used synthetic traffic, considering that injection rate of I/O operations is exponentially distributed and it is the same for all the servers attached to the storage network. Moreover, we have assumed that the destination disk of each I/O operation initiated by a server is randomly chosen among all the disks in the network. We have also assumed that the number of read operations is similar to the number of write operations. Finally, the size of control and data messages have been set to 32 and 2048 bytes, respectively.

In the second stage of our study, we have used real I/O traces in order to obtain more realistic results. The trace files we managed have been provided by Hewlett-Packard Labs [16]. These traces are representative of the workload typically found in office and engineering timesharing systems (simulation, compilation, editing and mail) and in file servers. The I/O traces contain information about the time when the I/O request starts, whether it is a read or a write operation, the requested data size, the selected disk, and the initial sector address.

Finally, disk access time has been shown to be the dominant factor in the total I/O operation response time [12], thus becoming the bottleneck of the entire storage system. In order to stress the network, we have assumed that disks access data fast enough to avoid becoming a bottleneck. This assumption is valid for I/O write operations because disks could initiate the response to the server once the information is written to the disk write cache. In the case for I/O read operations, it depends on the locality (e.g., sequentiality) level of the accessed information, that may be noticeable in multimedia or internet browsing loads.

3 Performance evaluation

As shown in the previous sections, in the design of a SAN there are a lot of parameters that must be taken into account, such as the locations for both servers and disks, network topology, switch architecture, routing algorithms, etc. Moreover, when evaluating the performance achieved using the different design parameters, some environment conditions must be considered, such as the pattern of the arrival traffic, message length, message destination distribution, etc. Also, some technology-dependent information must be taken into account, such as the time needed by the switch to route messages, the propagation delay along physical links, etc.

In order to accurately model this great number of parameters, we have used byte level simulation. Our SAN simulator [12] has been implemented using the CSIM language [6]. CSIM consists of a library of procedures, functions, and macros which give C (and C++) programmers a powerful tool for developing discrete-event, process-oriented simulation models. A CSIM program models a system as a collection of CSIM processes which interact with each other by using internal structures to CSIM. The SAN simulator has been written in ANSI C code in order to enable system portability, and it has about 10,000 lines.

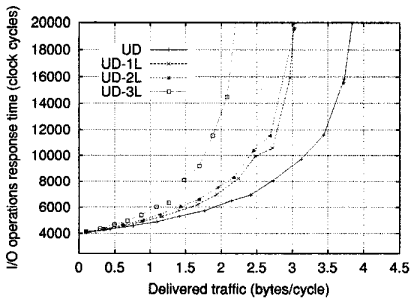
Simulations were run for a number of cycles high enough to obtain steady values of I/O operation response times. When the network is close to saturation, simulations were run for a number of cycles high enough to deliver 25,000 I/O operations (50,000 messages), that is, a total amount of about 50 MB of information transmitted between servers and disks. When using real I/O traces, the simulation stopped when 25,000 I/O operations were transmitted.

In the following sections we present an analysis of the results obtained by simulation. For the sake of simplicity, we will refer to the up*/down* and minimal adaptive routing algorithms without failures as UD and MA, respectively. A suffix like nL will indicate n link failures in the network. Although we have studied three different topologies for each network size, only results for the most representative of them are presented.

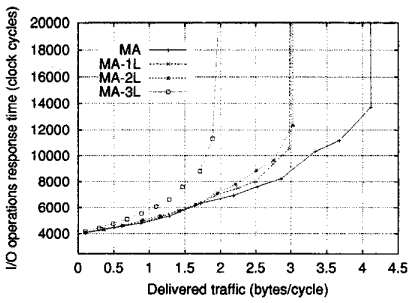
3.1 Synthetic I/O Load

Figure 4(a) shows the effect of link failures on network performance when the first switch architecture implements the up*/down* routing algorithm in a network composed of 8 switches. As can be seen, as links fail, performance is worsened. The loss in connectivity increments the I/O response time at the same time that network throughput is decreased. Due to topology characteristics, the failure of a second link slightly affects network performance. However, when the third link fails, achieved throughput is half the network throughput obtained when all the links are properly working. Note that three faulty links represent a loss of 25% the available connectivity. When network size is 16 switches, such a number of failures represents much fewer percentage of the total network connectivity (12.5%), and therefore, the decrement in network performance is less noticeable, as can be seen in Figure 4(c). Surprisingly, the failure of two and three links slightly increment throughput with respect to the case when no link has failed. The reason for this is that in this network topology, when two or three links fail, the routes used by messages to reach their destinations are shorter, due to better selection of the root switch. This is an example of the well known problem related with up*/down* routing, which in most cases is not able to provide minimal routing, and also concentrates network traffic near the root switch.

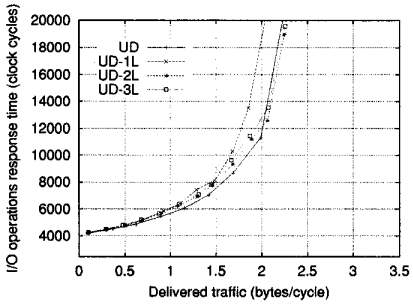
Figure 4(b) shows how link failures affect network performance when the first switch architecture uses the minimal adaptive routing scheme in a 8-switch network. Note that achieved throughput in the absence of link failures is slightly higher than in the case for the up*/down* routing algorithm. This is due to the fact that the minimal adaptive routing algorithm provides more minimal routes than the up*/down* routing scheme. Therefore, as links fail, besides losing network connectivity as in the case for up*/down* routing, the minimal routes also increase their length, thus decreasing network performance, as can be seen in Figure 4(b). This fact makes the minimal adaptive routing scheme more sensitive to failures than the up*/down* algorithm. This effect is more noticeable in larger networks, as shown in Figure 4(d). Note that in this case, network



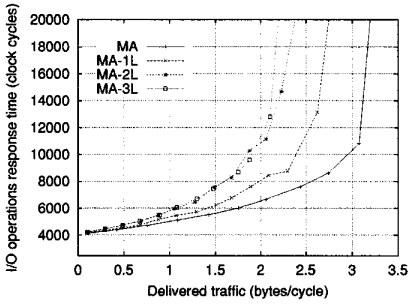
(a) 8 switches, up*/down* routing



(b) 8 switches, minimal adaptive routing



(a) 16 switches, up*/down* routing



(b) 16 switches, minimal adaptive routing

Figure 4. Effect of link failures when using the switch architecture I

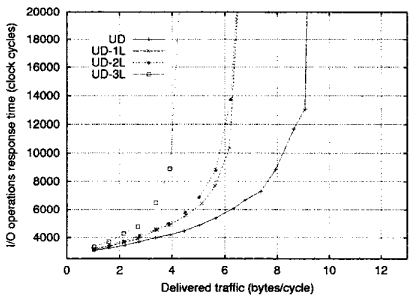
throughput achieved by MA is noticeably higher than the one obtained by UD, because in a larger network differences between the minimal routes and the ones provided by the up*/down* scheme are more significant. As can be seen in Figure 4(d), as links fail, the performance of the minimal adaptive algorithm noticeably decreases, especially for the first failure. Note that the increment in performance of the up*/down* scheme for two and three link failures causes a lower decrement in the performance of the MA algorithm. Also note that although the minimal adaptive scheme is more sensitive than the up*/down* one to link failures, network performance remains higher. In the case for larger networks, MA performance when three links have failed is even higher than the performance of the UD scheme in the absence of faulty links.

When Fibre Channel switches implement the architecture II, revisited in Section 2.1, the effect of link failures is presented in Figure 5. Figure 5(a) shows the case for a 8-switch network when the up*/down* routing algorithm is used. As can be seen, network performance when this second architecture is used is almost twice the one obtained by the previous architecture. As in the case before, as the number of faulty links increments, the overall performance is decreased. However, even in the case for three link failures, network performance is higher than in the case for the first architecture in the absence of faulty links. When larger networks are used, similar effect to those presented in Figure 4(c) are obtained.

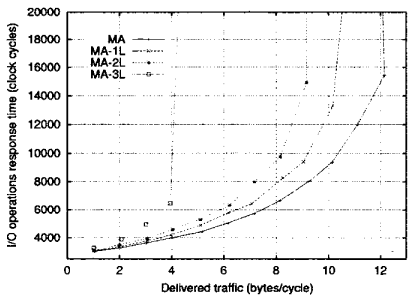
When the minimal adaptive routing is used in conjunction with the second switch architecture, the effect of link failures on network performance is shown in Figures 5(b) and (d) for 8 and 16-switch networks, respectively. In this case, network throughput is three times higher than the one obtained by architecture I, as can be seen comparing Figures 4(b) and (d) with Figures 5(b) and (d), respectively. As in the case before, as links fail, network performance is decreased due to both the lower aggregated bandwidth and the greater length of minimal paths. However, network performance obtained by the MA algorithm is always higher than the one obtained by the UD scheme.

3.2 Real I/O Load

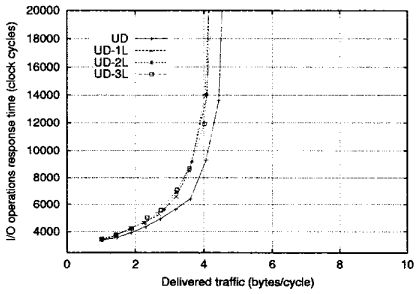
In the previous section we have stressed the network by using synthetic traffic. This traffic model allows us to easily evaluate the full range of network load. However, in most cases it may differ from the traffic present in actual systems. In order to complete our analysis of the effect of link failures on network performance, in this section we will stress the network by using real I/O traces [16]. To do so, we will apply the traces in two different ways: moderate and high network loads. In the first one, we have used the traces in a relaxed fashion, that is, we have scaled them in a way that network load has been lowered. In the second one, we have



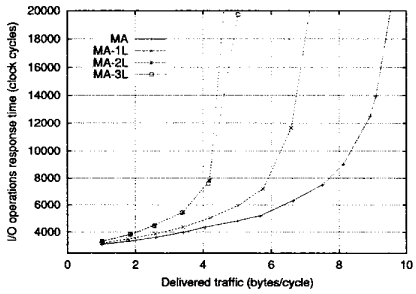
(a) 8 switches, up*/down* routing



(b) 8 switches, minimal adaptive routing



(a) 16 switches, up*/down* routing



(b) 16 switches, minimal adaptive routing

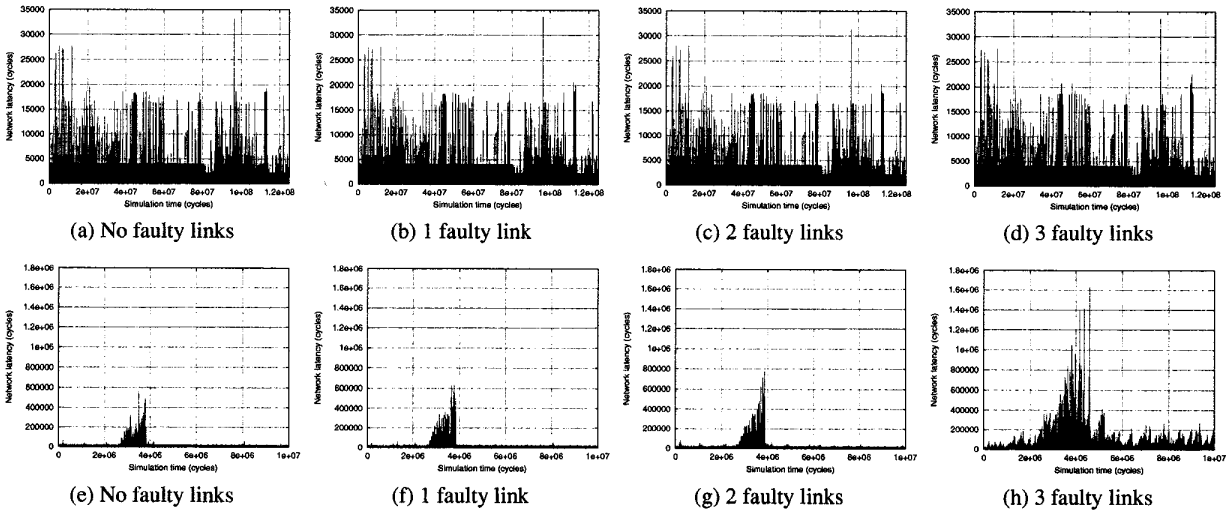
Figure 5. Effect of link failures when using the switch architecture II

compressed the whole trace in order to make network load ten times higher than in the case for low load. Although not shown, the overall I/O traffic injected into the interconnection network for both moderate and high loads is bursty, presenting long periods where injected traffic remains relatively low, and some short intervals in which network traffic considerably increases. In this section we will focus in the case for a network composed of 8 switches implementing the second of the architectures presented in Section 2.1.

Figures 6 and 7 display message latency for data and control messages, respectively, for the 8-switch network when the up*/down* routing scheme is used. Both scenarios, moderate and high load, are plotted. As can be seen, in the case for low I/O load, the presence of faulty links in the network has no significant effect on the overall performance. This is due to the fact that available network bandwidth is able to deal with such a load, even in the case when there are three faulty links in the network. Comparing these results with the ones presented in Figure 7(a), we can see that with this moderate load, the network becomes saturated only for some short periods of time, that is, when network traffic achieves four or more bytes per cycle. Therefore, the network if able to properly absorb these short bursts. However, in the case for high I/O load (Figures 6(e)-(h) and 7(e)-(h)), simulation results are quite different. In this case, we can see that as the number of faulty links increases, the lower aggregated network bandwidth makes message latency to increase. As can be observed in Figures 6(e)-(h), data message latency significantly increases, especially in the presence of three faulty links, where the network remains saturated for the whole simulation, presenting message latencies in many cases higher than 100,000 cycles.

On the other hand, when minimal adaptive routing is used, simulation results are similar in the case for moderate I/O load, as shown in Figures 8(a)-(d) and 9(a)-(d) for data and control messages, respectively. In this case, the network is also able to manage such a moderate traffic, and thus differences in performance are negligible. However, when higher network loads are applied (Figures 8(e)-(h) and 9(e)-(h)), differences in performance arise. In the case for the minimal adaptive routing algorithm, one can observe that message latency is significantly lower than for up*/down* routing. Also, as the number of faulty links increases, message latency only increases in the periods of higher activity, remaining the base message latency for the rest of the simulation time. These results are very important, because they mean that although the minimal adaptive routing is very sensitive to link failures, as mentioned in the previous section, it is able to deal with bursty traffic even in the presence of severe adverse conditions, as is the case where three links are unavailable simultaneously.

Finally, although not shown, we have computed the cumulative distribution function for I/O response time and latencies of both data and control messages when us-



(a) No faulty links (b) 1 faulty link (c) 2 faulty links (d) 3 faulty links

(e) No faulty links (f) 1 faulty link (g) 2 faulty links (h) 3 faulty links

Figure 6. Network latency for data messages when using up*/down* routing: low load (a)-(d) and high load (e)-(h)

ing up*/down* and minimal adaptive routing, respectively. When moderate network load is considered, curves are similar for the two routing algorithms, meaning that there is no significant difference in performance, as seen before. However, in the case for high loads, the minimal adaptive routing algorithm behaves slightly better than up*/down* routing for a small number of link failures. Only in the case for three faulty links, differences are more noticeable, showing that the former manages better bursty traffic than the latter.

4 Summary

In this paper, we have analyzed the performance degradation of storage area networks due to the presence of link failures. The evaluation study was performed on networks with 8 and 16 switches, in order to consider both small and medium size configurations. In our analysis, we have used synthetic and real I/O traffic in order to evaluate the network performance when links fail of two different routing algorithms. We have also evaluated two different Fibre Channel switch architectures. Simulation results show that network performance degrades in the presence of failures, but this degradation depends on the routing algorithm, the network size and the switch architecture.

Synthetic traffic has pointed out that, for small networks, both up*/down* and minimal adaptive routing algorithms are sensitive to faulty links in a similar manner, increasing message latency and decreasing network throughput. However, as network size increases, the latter becomes more sensitive than the former due to the fact that the number of minimal routes is highly reduced. On the other hand, the effect of link failures on network performance depends on the switch architectures used, being more negative for the more efficient one.

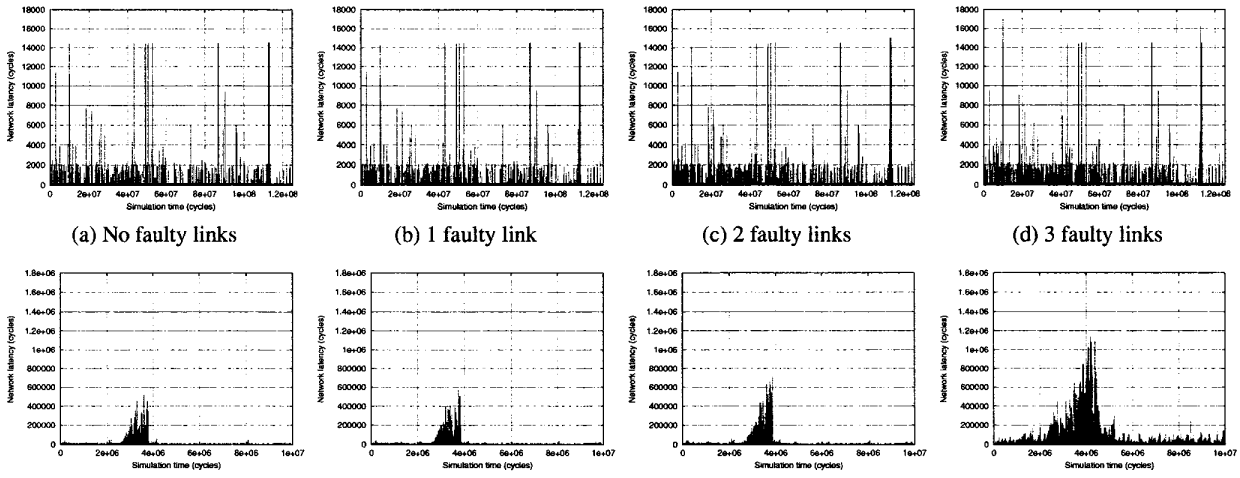
In order to obtain more accurate results, we have injected real I/O traffic into the network. Real I/O traffic is quite different from the synthetic one used, presenting long periods where the injected traffic remains relatively low, and also sharp bursts. Therefore, obtained results slightly differ from those for synthetic traffic. In this case, for moderate loads, the effect of link failures is not noticeable due to the low traffic itself, which can be efficiently absorbed by the network. However, for high loads, the minimal adaptive routing algorithm better deals with bursts, even in the presence of severe adverse conditions.

Acknowledgements

The authors thank Herb Schwetman from Mesquite Software, who helped us with the implementation of the SAN simulator, and John Wilkes and Richard Golding from HP-Laboratories, who provided the real-world I/O traces and motivated our interest in storage systems.

References

[1] N. J. Boden, D. Cohen, R. E. Felderman, A. E. Kulawik, C. L. Seitz, J. N. Seizovic and W. Su, *Myrinet - a gigabit per second local area network*, IEEE Micro, pp. 29–36, February, 1995.



(a) No faulty links (b) 1 faulty link (c) 2 faulty links (d) 3 faulty links

(e) No faulty links (f) 1 faulty link (g) 2 faulty links (h) 3 faulty links

Figure 7. Network latency for control messages when using up*/down* routing: low load (a)-(d) and high load (e)-(h)

[2] R. V. Boppana and S. Chalasani, *Fault-tolerant wormhole routing algorithms for mesh networks*, IEEE Transactions on Computers, vol. 44, no. 7, pp 848–864, July 1995.

[3] R. Casado, A. Bermúdez, F. J. Quiles, J. L. Sánchez and J. Duato, *Performance evaluation of dynamic reconfiguration in high-speed local area networks*, Proceedings of the 6th International Symposium on High-Performance Computer Architecture, pp. 85-96, January 2000.

[4] A. A. Chien and J. H. Kim, *Planar-adaptive routing: Low-cost adaptive networks for multiprocessors*, Proc. of the 19th International Symposium on Computer Architecture, May 1992.

[5] G. Ciardo, L. Cherkasova, V. Kotov and T. Rokicki, *Modeling a fibre channel switch with stochastic Petri nets*, Proceedings of 1995 ACM SIGMETRICS and PERFORMANCE'95, pp. 319–320, May 1995. (Full paper: HP Laboratories Report no. HPL-94-107).

[6] *User's guide: CSIM18 Simulation Engine (C version)*, Mesquite Software, Inc.

[7] B. V. Dao, J. Duato and S. Yalamanchili, *Configurable flow control mechanisms for fault-tolerant routing*, Proceedings of the 22nd Int. Symposium on Computer Architecture, pp. 220–229, June 1995.

[8] G. Ganger, *Generating representative synthetic workloads*, Proc. of the Computer Measurement Group Conference, pp. 1263–1269, December 1995.

[9] P. T. Gaughan and S. Yalamanchili, *A Family of Fault-Tolerant Routing Protocols for Direct Multiprocessor Networks*, IEEE Trans. on Parallel and Distributed Systems, vol. 6, no. 5, pp. 482-497, May 1995.

[10] M.E. Gómez and V. Santonja, *Self-similarity in I/O workload: analysis and modeling*, Workload Characterization: Methodology and Case Studies, IEEE Computer Society, pp. 97–104, 1999.

[11] P. Kermani and L. Kleinrock, *Virtual cut-through: a new computer communication switching technique*," Computer Networks, vol. 3, pp. 267–286, 1979.

[12] X. Molero, F. Silla, V. Santonja and J. Duato. *Modeling and simulation of storage area networks*, Proceedings of the 8th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems. IEEE Computer Society Press, August 2000.

[13] X. Molero, F. Silla, V. Santonja and J. Duato. *An improved switch architecture for implementing fibre channel storage area networks*, Submitted to the International Parallel and Distributed Processing Symposium. April 2001.

[14] X. Molero, F. Silla, V. Santonja and J. Duato. *Performance sensitivity of routing algorithms to failures in networks of workstations*, To appear in the proceedings of the 3rd International Symposium on High Performance Computing. Lecture Notes in Computer Science, October 2000.

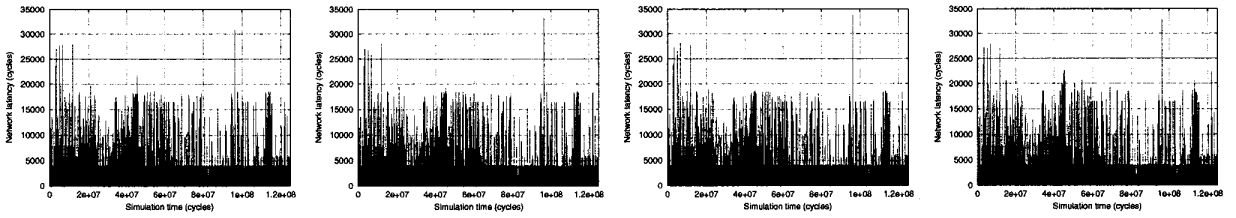
[15] B. Phillips, *Have storage area networks come of age?*, Computer, vol. 31, no. 7, pp. 10–12, July 1998.

[16] C. Ruemmler and J. Wilkes, *Unix disk access patterns*, Proceedings of the 1993 Winter USENIX Conference, pp. 228–235, 1993.

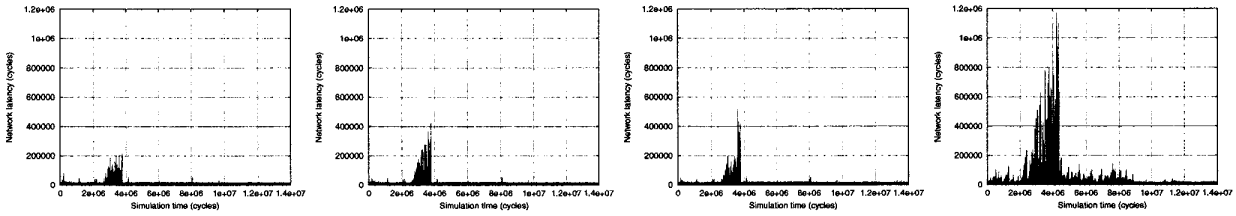
[17] M. D. Schroeder, et al. *Autonet: a high-speed, self-configuring local area network using point-to-point links*, Technical Report SRC research report 59, DEC, April 1990.

[18] F. Silla and J. Duato, *Improving the efficiency of adaptive routing in networks with irregular topology*, Proceedings of the 1997 International Conference on High Performance Computing, December 1997.

[19] D. Tang, *Storage area networking: the network behind the server*, Gadzoox Microsystems Inc., 1997.

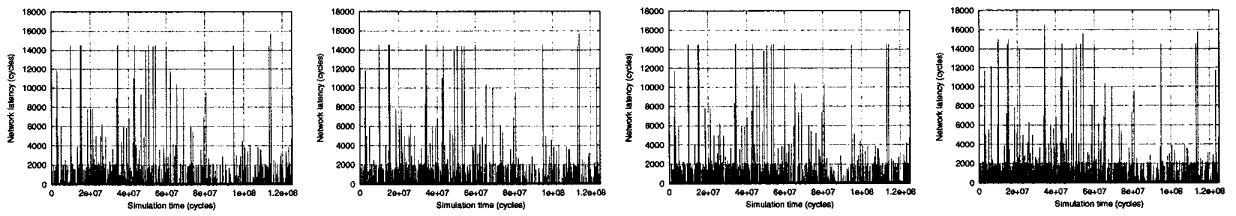


(a) No faulty links (b) 1 faulty link (c) 2 faulty links (d) 3 faulty links

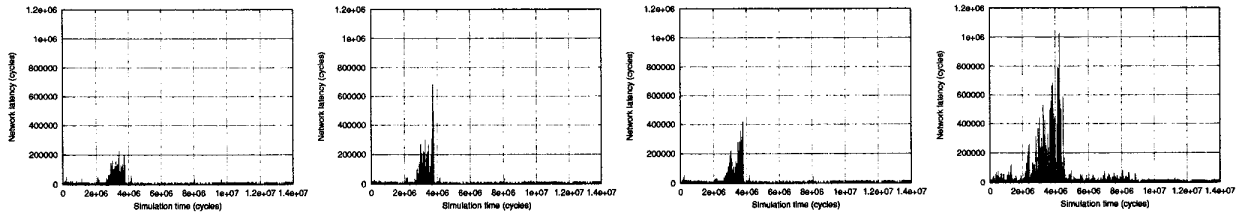


(e) No faulty links (f) 1 faulty link (g) 2 faulty links (h) 3 faulty links

Figure 8. Network latency for data messages when using minimal adaptive routing: low load (a)-(d) and high load (e)-(h)



(a) No faulty links (b) 1 faulty link (c) 2 faulty links (d) 3 faulty links



(e) No faulty links (f) 1 faulty link (g) 2 faulty links (h) 3 faulty links

Figure 9. Network latency for control messages when using minimal adaptive routing: low load (a)-(d) and high load (e)-(h)