Distrib. Syst. Engng 1 (1994) 332–344

A performance analysis of advanced I/O architectures for PC-based network file servers

Khoa D Huynh†§ and Taghi M Khoshgoftaar‡||

† IBM Corporation, Internal Zip 1430, PO Box 1328, Boca Raton, FL 33429-1328, USA
‡ Department of Computer Science & Engineering, Florida Atlantic University, Boca Raton, FL 33431, USA

Received 18 March 1994

Abstract. In the personal computing and workstation environments, more and more I/O adapters are becoming complete functional subsystems that are intelligent enough to handle I/O operations on their own without much intervention from the host processor. The IBM Subsystem Control Block (SCB) architecture has been defined to enhance the potential of these intelligent adapters by defining services and conventions that deliver command information and data to and from the adapters. In recent years, a new storage architecture, the Redundant Array of Independent Disks (RAID), has been quickly gaining acceptance in the world of computing. In this paper, we would like to discuss critical system design issues that are important to the performance of a network file server. We then present a performance analysis of the SCB architecture and disk array technology in typical network file server environments based on personal computers (PCs). One of the key issues investigated in this paper is whether a disk array can outperform a group of disks (of same type, same data capacity, and same cost) operating independently, not in parallel as in a disk array.

1. Introduction

As processor performance continues to increase at an enormous pace and advanced memory architectures allow the host memory to keep up with processor speed, the I/O subsystem in a computer is fast becoming the system bottleneck. Since it depends on either network media technology or mechanical devices with moving parts, the I/O performance has lagged behind the processor and memory speeds. The I/O performance problem becomes even more acute as large, expensive mainframe computers give way to a network of smaller workstations and personal computers. In a networking environment, the large-scale storage of shared information assumes paramount importance. In such an environment, hundreds of users, each with his or her own personal computer, can work and share gigabytes of programs, application packages, information databases, and I/O devices, such as laser printers, plotters, etc. Network file serving is therefore one of the most I/O-intensive applications for personal computers and workstations today.

Fortunately, there have been some major advances in the I/O area recently. More and more I/O adapters (controllers) are becoming complete functional subsystems that are intelligent enough to handle I/O operations on their own without much intervention from the host processor. To take advantage of these intelligent I/O adapters, IBM has proposed the Subsystem Control Block (SCB) architecture, which defines the services and conventions that deliver command information and data to and from the adapters. The SCB architecture has two operating modes, the Locate Mode and the Move Mode. The Locate Mode represents the conventional, interrupt-driven I/O protocol used in many current computers, while the Move Mode provides a true peer-to-peer I/O protocol between any two units in the system (the term 'unit' here refers to either an intelligent I/O adapter or the host processor in the system). In recent years, a new storage architecture, called the Redundant Array of Independent Disks (RAID), has also been gaining acceptance quickly in the world of computing. How do these advances in I/O adapters, I/O protocols, and disk organizations affect the overall performance of today's personal computers? In this paper, we focus on personal computers that are used as network file servers. Using simulation models, we consider two high-end personal computer systems, which are identical in all aspects—except for their I/O subsystems: one implements the RAID technology, and the other does not. We then evaluate the impact of these I/O subsystems, operating with either the conventional I/O processing protocol or the new peer-to-peer I/O protocol embodied in the SCB architecture, on the overall system performance.

§ E-mail: khoa@vnet.ibm.com
|| E-mail: taghi@cse.fau.edu

2. The Subsystem Control Block architecture

The Subsystem Control Block (SCD) architecture enhances the potential of intelligent I/O adapters by defining the logical protocols and control structures that are used to transfer command or control information, data, and status information between the host processor and an I/O adapter, or directly between I/O adapters themselves (peer-to-peer) [1]. The architecture provides command chaining, data chaining, signalling, and synchronization of commands and status information. It separates the delivery of control information from data delivery to increase the system performance, raise the level of functional capability, and provide more design flexibility. The SCB architecture has two operating modes: the Locate Mode and the Move Mode.

2.1. Conventional I/O protocol (SCB Locate Mode)

Many intelligent I/O adapters used in personal systems today support the Locate Mode (or some variants) of the SCB architecture. It is an interrupt-driven I/O protocol. In the Locate Mode, the control structure is a relatively fixed format structure, called the Command Control Block. This structure allows the command, control, and status information, as well as pointers to data buffers in host memory, to be passed from the host processor to an I/O adapter (figure 1).

In the Locate Mode, only the host processor can send requests to an I/O adapter. To send a request to an I/O adapter, the host processor first builds the control block in host memory to describe the I/O operation that needs to be performed (an I/O request). It then writes the physical memory address of the control block to the adapter's command interface registers, and a device identifier to the adapter's attention register, which interrupts the adapter's on-board processor. After being interrupted, the adapter's processor uses the memory address in its command interface registers to locate the control block in host memory and fetches it into its own storage area for execution. After the I/O request is completed, the adapter interrupts the host processor, and supplies it with status information. The adapter's reply to the host processor must be synchronized with the request.

The operation of the Locate Mode is serial in that the host processor cannot send another request to the same I/O device (disk, LAN connection, etc) until the current request has been completed. There can only be one request active per device at any given time, because requests to the same device cannot be 'tagged'. The host processor, however, can send requests to other devices through the same or a different I/O adapter. One exception is the IBM

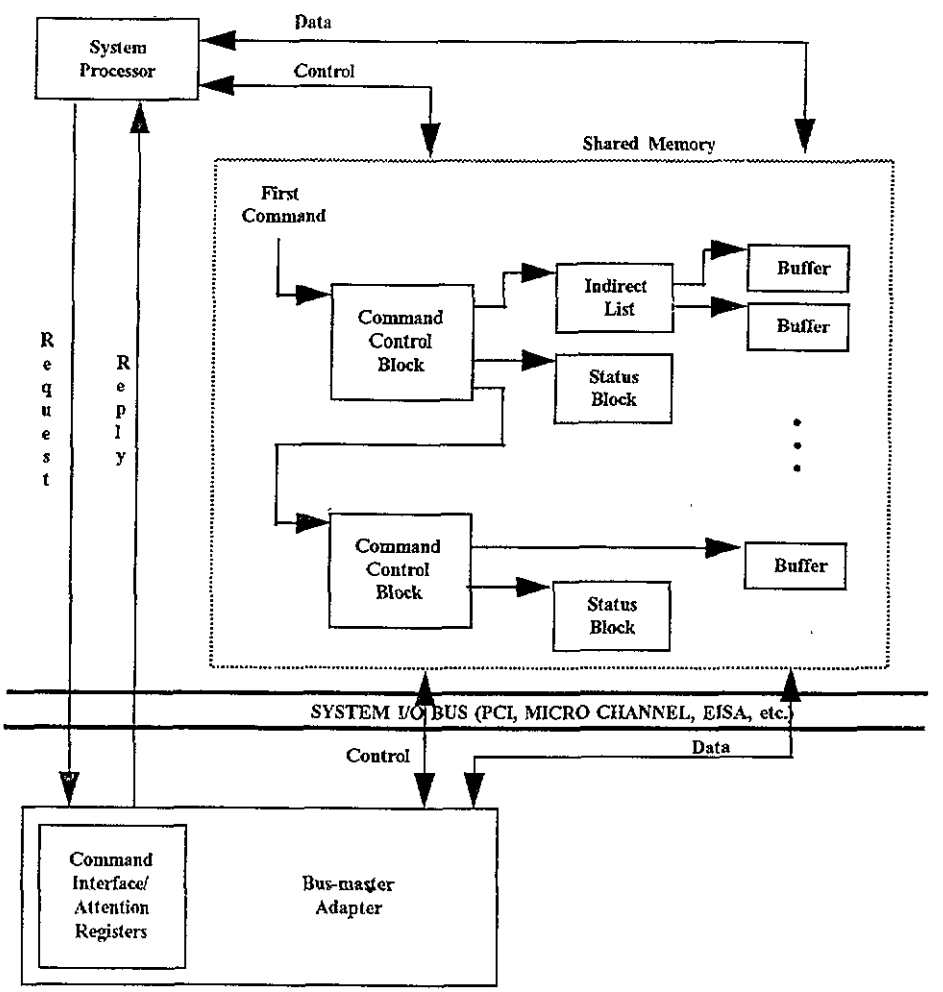

Figure 1. An interrupt-driven I/O architecture (SCB Locate Mode).

PS/2 RAID adapter. The main difference between the SCB Locate Mode and the RAID's I/O protocol is that the latter can support the queuing of multiple requests (up to 61) in the adapter's memory buffer. This is necessary because the whole disk array is treated by the system as a single logical disk unit.

2.2. Peer-to-peer I/O protocol (SCB Move Mode)

The Move Mode supports I/O data transfers by using shared memory interfaces to deliver requests and control-related information between any two units in the system (figure 2). The term 'unit' here is used to indicate an intelligent I/O adapter or the host processor. This provides the true peer-to-peer relationship between all system components. The key feature in the Move Mode is that an I/O adapter can send requests to, and accept requests from, another I/O adapter, not just the host processor. In contrast, in the Locate Mode, an I/O adapter can only receive requests from the host processor. At the present time, only the IBM PS/2 SCSI-2 adapter can support the SCB Move Mode.

The Move Mode uses control elements instead of control blocks. The control elements are variable in length and can contain I/O requests, status, or error notifications. It is used by one unit to deliver a request or reply to another unit. The control elements are moved between each pair of units in the system through a pair of delivery pipes. Each pipe behaves as a FIFO queue, and allows for the delivery of control elements in only one direction. Full-duplex operation thus requires a pair of pipes. There is one pair of delivery pipes for each pair of units that want to communicate with one another. Figure 2 shows three pairs of delivery pipes used by two I/O adapters and the host processor to communicate with one another. A delivery pipe must be in memory shareable by the sending and receiving units. This does not mean that the pipes must reside in host memory; they can be in an I/O adapter's memory buffers, provided that any other adapter or host processor who wants to communicate with it can access those memory buffers. The

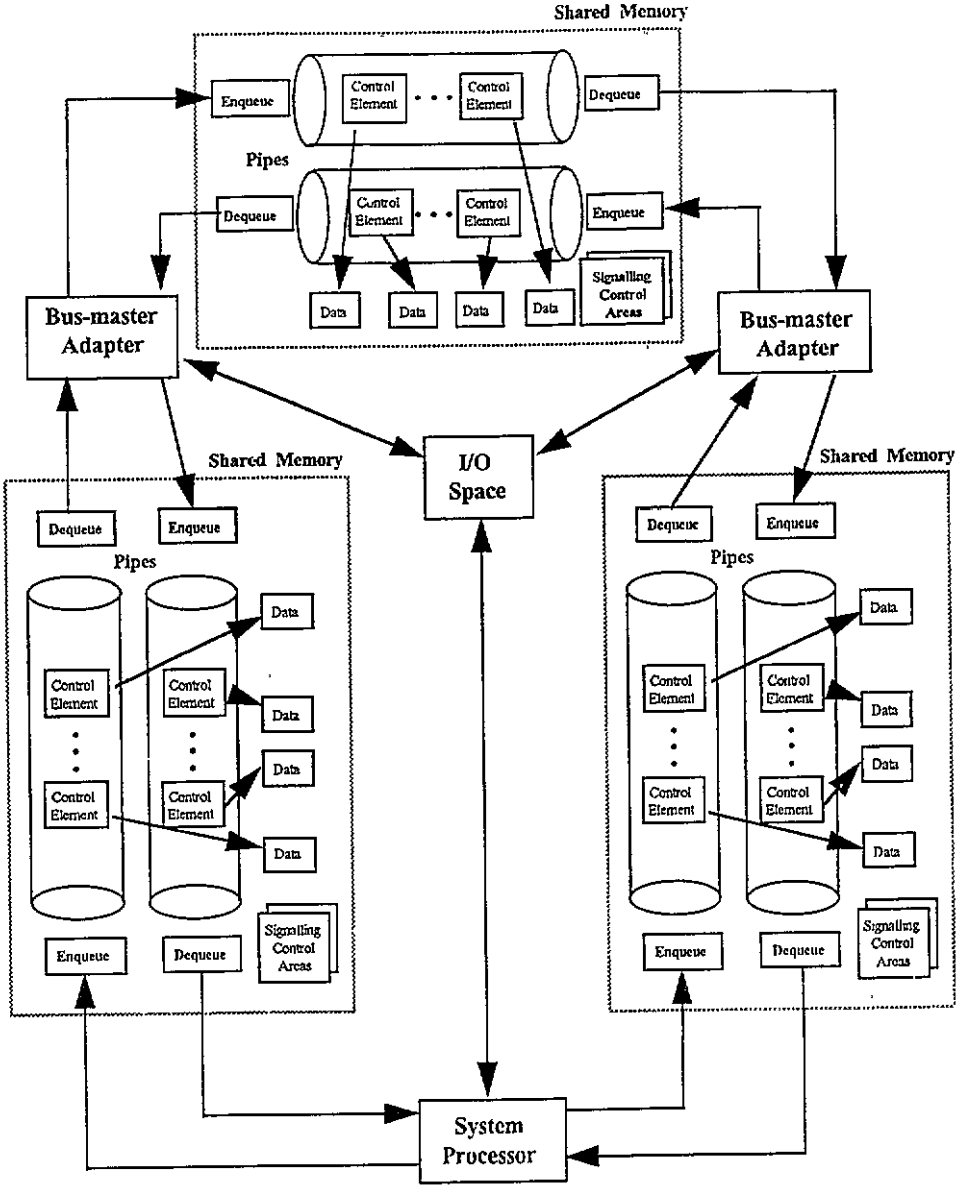

Figure 2. A peer-to-peer I/O architecture (SCB Move Mode).

pipes allow multiple control elements to be queued and processed asynchronously by each adapter or host processor. Unlike the Locate Mode, the Move Mode is not serial in nature (it can support the queuing of many requests to the same I/O device in the delivery pipes) and requires a much smaller amount of interrupts.

In the case of large sequential data transfers, such as a video stream, the Move Mode is very effective. In this case, the host processor running the operating system must initialize one of the I/O adapters (say, the token-ring adapter) at the beginning of each video stream transfer. After that, the host processor and host memory can be used for other work while the sequential video stream is transferred directly between the SCSI and token-ring subsystems.

3. Systems under consideration

We would like to evaluate how the SCB Locate Mode and Move Mode can affect the performance of network file servers equipped with either a disk array (RAID) or a number of independent disks. But first, we need to discuss the network file server environments that are selected for our study. We assume a high-end personal computer, like the IBM Personal System/2 Model 95 XP 486 50 MHz Server (or just 50 MHz Server for short), as the server system in this study.

3.1. Application environments

The workload characteristics of network file servers vary widely, dependent on the installations. Each LAN installation has a different number of users on the network, different machines, different shared I/O devices, and different types of shared information. Additionally, the workload characteristics of a given LAN installation also depend on the time of usage: the network workload is usually light on weekends and evenings, and becomes heavy during the afternoons. In order to evaluate the performance impact of different I/O architectures on network file servers, we have designed several workloads that are based on a number of studies [2–6]. According to those studies, most network environments have the number of reads three times as high as the number of writes, and we use this 3 : 1 read-to-write ratio for all workloads used in our study.

University workload. This workload is based on a study performed at the University of Saskatchewan [2]. This distributed computing environment forms part of the Computer Science Research Laboratory in the Department of Computer Science, and is based on a 10 Mbps Ethernet network. The machines on the network consists of 14 Sun 3/50, one Sun 3/60, one MIPS M/120, and 10 HP 9000/340 series workstations. The server system is a Sun 3/180, providing user file access to more than 250 potential users, comprising faculty, staff, and students in the Department. The MIPS system averages 17 users during prime time hours. The Sun and HP workstations are usually occupied by only one user at a time. Some HP systems are diskless. All operating systems are UNIX-based, and remote file service is provided by the Network File System (NFS). There is a large amount of text processing, programming development activities (editing, compiling, debugging, etc), and e-mail processing. As such, the workload in this environment is typical for many academic environments. In our study, we assume the IBM PS/2 Model 95 50 MHz Server is the file server instead of the Sun 3/180 system. The request size distribution used in our models for this workload is based on the results reported in [2]. The request interarrival time, as measured in [2], is a hyper-exponential distribution with a mean of 0.80 seconds and a coefficient of variance of 2.98. The average file system cache hit ratio is assumed to be approximately 60%, which is typical for most network file server environments [6].

Medium workload. The request size distribution of this workload is similar to that of the university workload. The average file system cache hit ratio is still 60%. However, the request interarrival time distribution is based on [5], which describes a packet-train model for LAN traffic. There is an intra-train distribution and an inter-train distribution. The time intervals between 'trains' of packets are very large (in the order of tens of seconds), so we do not really care about them. Within each 'train', however, the packets arrive in close succession, so it is more interesting to see how the server handles these packets. As a result, we are more interested in the intra-train time distribution. In the models, we use the hyper-exponential distribution with mean 0.050 seconds and coefficient of variance 1.6, as measured in [5], for request interarrival time distribution.

Heavy workload. To see how different I/O architectures behave in a heavy network file server environment, we came across a study performed at the University of California at Berkeley (UCB) [4]. The workload measured and reported in this study is much heavier than previous studies. The measurement environment has a single-cable Ethernet in the Computer Science Department at UCB, which connects about 100 machines: 41 diskless Sun workstations; 23 XEROX Star workstations; three VAXs, several MicroVAXs; and a few Symbolics and TI Explorer LISP machines. Two of the VAXs and a Sun workstation are used as gateways to other UCB networks. User activities include program development, text editing, operating-system research, and experimentation with compute-intensive programs. This is a typical environment in a large research organization. We assume in our study that all 100 machines share our PS/2 50 MHz Server on a 16 Mbps token-ring network (instead of an Ethernet). The request interarrival time distribution is based on the cumulative percentage of request arrivals reported in the UCB study, which notes that 50% of the requests are followed by the next request within 3 ms, 84% are followed within 10 ms by another, and 99% within 90 ms. The packet sizes on the 10 Mbps Ethernet are different from those on the 16 Mbps token

ring, so we assume the request size distribution in this case is the same as the one in the university environment discussed earlier. As in previous workloads, we assume that the average file system cache hit ratio is approximately 60%.

Very heavy workload. We have decided to go one little step further with the UCB-based workload mentioned here. We assume that there are so many machines on the network doing different tasks that the file system cache hit ratio on our PS/2 50 MHz Server drops to 20%. This low cache hit ratio on the file server can be caused by the fact that some client machines have their own file system caches for disk I/O. The presence of these client caches reduces the traffic to the file server, but it also increases the cache miss ratio on the server [6]. However, in our case, we assume that we have so many more machines on the network that the request interarrival time distribution is not changed from the UCB-based workload. In other words, the very heavy workload is the same as the heavy workload discussed in the preceding section except that the file system cache hit ratio on the server drops to 20% (from 60%).

3.2. Operating system environment

The operating system maintains the file system cache. In our models, we assume a 4 MByte file system cache, with a cache block size of 8 Kbyte. These sizes are very common on network file servers. A disk read request that is less than 8 Kbyte and not in cache (read cache miss) will cause the entire 8 Kbyte cache block to be read in from disks. A write cache miss will not cause anything to be brought into the file system cache. The models also assume a small 128 Kbyte write buffer. Disk write requests are not written to disks immediately. First they are written to the write buffer, and at the first opportunity that the disk I/O subsystem is not busy, they will be written to disks. If the write requests are to the same track or cylinder on the same disk, they can be combined together into a single request, thus reducing the disk write traffic. According to a trace-driven analysis of write caching policies for disks [7], a small 128 Kbyte cache can reduce the disk write traffic by 10%, which is used in the models.

3.3. Hardware environments

We consider the IBM Personal System/2 (PS/2) Model 95 50 MHz Server, as shown in figure 3, as our server platform. It has the usual components of a personal computer system, such as the host processor, host memory, a SCSI I/O disk subsystem, a token-ring subsystem, and a central system bus (the Micro Channel). For the high-end PS/2 50 MHz Server, the host processor is a 50 MHz i486DX processor with 256 Kbyte of second-level cache. In some cases, we also consider a mid-range PS/2 Model 70 system with a 25 MHz i386DX processor and 64 Kbyte of external cache. In our work, we assume that the amount of physical memory is quite large so that we would never reach the memory overcommit state; there is no memory swapping as a result of memory shortage. However, we do take into account the memory access contention in our models. The memory access speed is dependent on the host processor's cycle time. The PS/2 50 MHz Server uses the dual-path design for its memory controller. This design implements separate data paths and FIFO queues for the host processor and the Micro Channel. This enables both the host processor and the Micro Channel to buffer memory requests simultaneously and perform other tasks while the memory controller processes those memory requests queued up in the FIFOs.

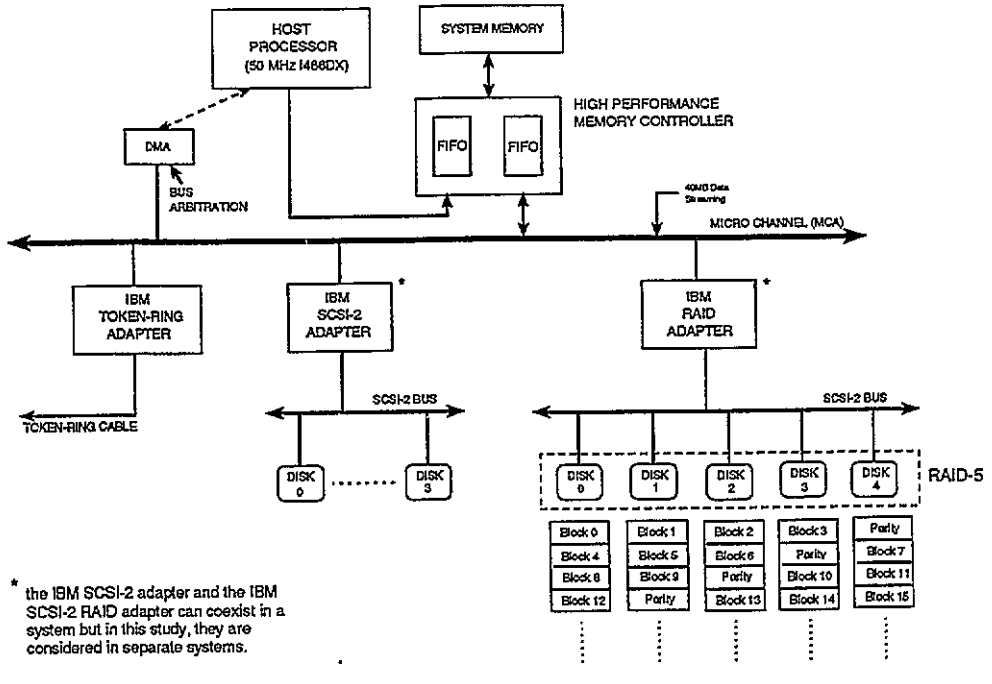

Figure 3. The IBM PS/2 Model 95 50 MHz Server.

System I/O bus (the Micro Channel). The high-end PS/2 50 MHz Server supports the 32-bit data streaming mode of the Micro Channel, with an effective data transfer rate of 40 Mbytes/s (32-bit data bursting on the channel with 100 ns cycle time). If there is another transfer waiting, the current transfer has a maximum of 4 μs to stream data across the Micro Channel before having to relinquish control of the channel. The average arbitration overhead on the Micro Channel is approximately 10% of the data transfer overhead. The mid-range PS/2 Model 70 has a 32-bit Micro Channel with 300 ns basic cycle time, giving an effective data transfer rate of 13.33 Mbytes/s.

SCSI-2 I/O subsystems. As shown in figure 3, the disk I/O subsystem on the PS/2 50 MHz Server implements the ANSI SCSI-2 standard, which allows multiple commands to be queued to an I/O device (supporting the SCB Mode Move) and specifies a maximum data transfer rate of up to 20 Mbytes/s.

We consider two different SCSI-2 I/O subsystems: one implements the RAID disk array technology, and the other does not. We do not attempt to compare the disk array performance to a single large disk (SLED). Many other performance studies already did RAID versus SLED comparisons [8], and found that, unsurprisingly, disk arrays are indeed better in performance than SLEDs. What we really want to find out in this study is whether a disk array can outperform a group of disks (of same type, same data capacity, and same cost) operating independently, not in parallel as in a disk array.

Non-RAID I/O subsystem. In a non-RAID disk configuration with multiple disks, there is no data striping and the disks operate independently. A single data file must reside on a single disk. For this I/O subsystem, we use the IBM PS/2 SCSI-2 adapter, which is the only commercially available adapter that can support both the SCB Locate and Move Modes. The IBM SCSI-2 adapter has an Intel 80C186 microprocessor to handle the SCB protocols and control the data transfers. It also has a small dual-ported buffer to handle data transfers to and from the Micro Channel at an effective data rate of 40 Mbytes/s. At the other end of the adapter is one 16 bit, 20 Mbytes/s SCSI-2 bus (in our study, we assume only one SCSI-2 bus, as it is not a potential system bottleneck and each bus can handle up to 7 disks).

RAID I/O subsystem. A RAID I/O subsystem allows a single data file to be striped, or interleaved, across multiple disks (a disk array). This allows disk requests to be overlapped and processed in parallel by multiple disks in the disk array. Even though a simple disk can outperform a single large disk, it suffers in reliability. Many disks operating in parallel, where each disk contains only a portion of a data file, creates a system whose reliability is unacceptable. The solution to this reliability problem is to add redundancy to the disk array. The way in which a disk array interleaves data and adds redundancy determines the trade-offs between cost, performance, and data protection. The computer scientists at the University of California at Berkeley [9] defined several possible RAID implementations. The most popular configuration is RAID-5, where the data is striped by blocks, rather than bytes, across multiple drives. The block size at which data is striped, also called the interleaving depth, depends on the hardware implementation. Another special feature of RAID-5 is the fact that the parity information is distributed across all disks in the array, as shown in figure 3. If the parity blocks are on the same disk, then it would be RAID-4. The distribution of parity information among all disks in a RAID-5 array minimizes the performance bottleneck caused by excessive accesses to the parity disk in RAID-4 [9]. We have decided to focus on RAID-5 because of its tremendous popularity over other RAID configurations: virtually all existing disk array controllers, including the IBM PS/2 RAID adapter, support RAID 5. Very few, if any, are supporting RAID 3. RAID 0, 1, 2, and 4 configurations are not very practical for video or network file servers that are based on personal computers. In this paper, we assume a RAID-5 disk array with five disks. It should be noted that a five-disk RAID 5 system provides a 4-disk logical data capacity, excluding the parity information.

We have also decided to use an IBM disk array controller for two reasons. First of all, it employs a high-performance, cost-effective RAID-5 disk array implementation. Secondly, we have easy access to the architectural information and empirical performance data obtained from the hardware development groups here at IBM. This is necessary for model development as well as model validation. The IBM PS/2 RAID adapter has the following characteristics (which are all simulated in our models).

- Use an Intel i960 RISC processor for on-board processing. This processor is much faster than the 80C186 microprocessor used on the IBM PS/2 SCSI-2 adapter. However, it also has to execute more microcode in a RAID implementation, especially the calculation of parity information during write operations.

- Have up to 4 Mbytes of on-board memory buffer. About 1 Mbyte of this buffer is used to store microcode and other information pertinent to the operation of the adapter, so only 3 Mbytes are available for staging data coming to and from the disk array, as well as caching frequently used data. The buffer's cache block line is 8 Kbytes, so for requests larger than 8 Kbytes, this buffer cache behaves just like a staging buffer for I/O data moving between the Micro Channel and the disk array. For small read requests (smaller than 8 Kbytes) that are not in cache, the adapter will read 8 Kbytes from the disk array to fill in the entire cache block in the adapter's memory buffer.

In effect, for disk read operations, we have two levels of caching: the file system cache and the RAID adapter's memory buffer. We assume that the file system cache is fairly effective in reducing the read traffic, so that any read miss from the file system cache will likely miss the adapter's buffer cache as well. We assume that the RAID adapter's memory buffer can reduce the read traffic to the

disk array by about 20%. This 20% hit ratio for the adapter's buffer is derived from simulation studies in [6], and is based on the size of the file system cache considered in our study as well as the 3 Mbyte size of the adapter's memory buffer available for caching. However, the adapter's buffer can reduce the disk read traffic required for parity calculation during write operations by a much higher percentage. This is because the reads required for parity calculations are initiated by the adapter's processor, and thus, do not go through the file system cache. Our models assume that the adapter's buffer cache hit ratio for these special reads is the same as the file system cache's hit ratio (60%).

- Supports an interleaving depth of 8 Kbytes, which means that the data is striped in blocks of 8 Kbytes across all disks in the array. The collection, in logical order, of these 8 Kbyte blocks from the first drive of the array to the last drive of the array is called a *stripe*. For a disk read request of 8 Kbytes or less, only one disk in the array needs to be accessed. Since the data are interleaved, multiple small reads (8 Kbytes or less) can be overlapped and serviced simultaneously by different disks in the array. A read request for more than 8 Kbyte would require multiple disks to be accessed in parallel. For disk write requests, things get more complicated as parity information must be recalculated. Before a write can be performed to certain disk(s), the data within the same stripe must be read into the adapter's memory buffer from the remaining disks in the array for parity recalculation. These reads can be done in parallel. The adapter's i960 processor then recalculates the parity. After the recalculation, the data to be written for the write request and the new parity information must be written back to the disk array (in parallel).

- Does not support synchronized spindles for all disks in the array in order to allow multiple disks to process different small requests (8 Kbytes or less) concurrently. As a result, there is a performance penalty for large requests that require multiple disks to be accessed in parallel [10].

- Supports 32-bit data streaming on the Micro Channel at 40 Mbytes/s (as with the non-RAID IBM PS/2 SCSI-2 adapter). In our work, we assume that all five disks in the RAID-5 array are attached to the same SCSI-2 bus, as in the case of the non-RAID I/O subsystem. During a large disk read (more than 8 Kbyte), data coming from multiple disks must be merged in the adapter's memory buffer before going out to the Micro Channel.

SCSI disks and disk layouts. In our models, we consider the current state-of-the-art SCSI disks, such as the IBM 540 Mbyte, 3.5″ SCSI disks. Each SCSI disk used in the models spins at 6000 rpm, has an average seek time of 10 ms and raw media access speed of 3.5 Mbytes/s. Each disk has an on-board, 256 Kbyte read-ahead buffer. The buffer can be divided into 4, 8, or 16 segments; each segment can perform read-ahead operations for a different process (user). Since the disk requests in a network file server environment are rather random in nature, it is very difficult, if not impossible, to come up with optimal disk layouts. In our work, we do not make any assumptions about the disk layouts for a network file server.

- *Disk seek times.* The average seek time is 10 ms for completely random requests. However, the disk requests in a network server environment are not completely random. Some trace studies [3, 6] showed that the network file accesses exhibit a strong locality, which is why file system caches are so effective at reducing the read traffic going to the disk I/O subsystem. Consequently, in our models, we use the exponential distribution for disk seek times with an average of 8 ms in the network file server environment (for both RAID-5 and non-RAID disk configurations).

- *Disk rotational latencies.* In the network file server environment, most of the requests are small in size and are not sequential, so they cannot benefit much from the disks' read-ahead buffers. At 6000 rpm, it takes 10 ms per revolution, so we use the uniform distribution with range [0, 10 ms] to represent the disk rotational latencies in this environment.

- *Disk transfer rates.* For disk write operations, data must be written to the disk media, so the transfer rate in this case is 3.5 Mbytes/s. For disk read requests, the transfer rate can be a little bit higher because of the disks' read-ahead buffers. While not very effective for non-sequential reads, the disks' read-ahead buffers can slightly reduce the number of I/Os to the disk media due to the locality of file references. Extrapolating data from [3], we assume that the disk buffer hit ratio is about 30%, yielding an effective data transfer rate of approximately 5 Mbytes/s.

The token-ring subsystem. A 16 Mbps token-ring network, connected to the IBM Token-Ring bus master in the 50 MHz Server, is used in the models. It should also be noted that most token-ring network support products allow a maximum packet size of 8 Kbytes, so large requests will be broken down into 8 Kbyte packets for transmission on the token ring.

4. Modelling methodology

In our modelling work, we used the IBM Research Queuing Package (RESQ) [11]. We have decided to use the simulation approach mainly because the I/O architectures and the Personal System/2 system hardware modelled here are highly sophisticated, and thus, analytical solutions could not be easily obtained. In addition, the models need some capabilities that are only provided with simulation (not analytical modelling) in RESQ, such as the capability to make routing decisions based on the status of simulation conditions as well as probabilities, etc.

Low-level performance data for I/O devices were obtained using analytical methods. Operating systems' code path lengths were obtained empirically by using the DEKKO software analysis tool running on the IBM Operating System/2 Version 2.1. The overhead of

microcode executed on the I/O adapters was also empirically measured. Finally, all of these data were used as input parameters for a set of RESQ simulation models designed to obtain the overall system performance measurements. These models are capable of simulating all critical operational and performance characteristics of the system components, from the file system cache maintained by the operating system to the intricate disk access operations in a RAID-5 disk array.

We have validated the SCB Locate Mode models by empirical performance measurements collected on the real hardware. The scenarios used in these empirical measurements are artificial, but they are controlled and reproducible. The simulation results obtained from the models for those 'test' scenarios are always within 10% of the empirical data. The RAID-5 disk array models have been validated by visual animation (a graphical tracing facility provided in RESQ) for up to 100 consecutive disk I/O operations, and have been confirmed by some empirical data obtained for 'test' scenarios on an early prototype (the production-level IBM PS/2 RAID adapter is not available for our model validation efforts). There is no existing hardware that can support the SCB Move Mode (direct data streaming between two I/O adapters), so it is much harder to validate the Move Mode models directly. However, the same modelling techniques and many modelling assumptions used in the Locate Mode models are applied to the Move Mode models, so we are very confident about the Move Mode models.

5. Performance results

As we analyse the data obtained from the models, we first focus on the utilization of the main system components, such as the Micro Channel, the host processor, the I/O subsystems, and I/O devices. Then we look at the average overhead for I/O requests.

System I/O bus (Micro Channel) utilization. Figure 4 shows that the Micro Channel utilization in the SCB Locate Mode and RAID-5 systems is higher than that in the Move Mode. This is because both the Locate Mode and RAID-5's I/O protocol use the host memory as the intermediate destination for data transfers between the disk I/O subsystem and the token-ring network. In other words, each data transfer is two-legged: from source to host memory and from host memory to destination. In the Move Mode, there is no intermediate destination (the host memory is not involved). Each data transfer is one-legged: it takes place directly between the disk I/O and the token-ring subsystems. We would expect that the Micro Channel utilization in the Locate Mode and RAID-5 systems should be twice as high as in the Move Mode because of their two-legged data transfers as compared to one-legged data transfers in the Move Mode. However, this is not the case mainly because of the presence of the file system cache (for read requests) and the write buffer (for write requests) in the Locate Mode and RAID-5's I/O protocol. Both the file system cache and the write buffer reduce the traffic to the disk I/O subsystem (refer to section 3 for more details), and thus, reduce the traffic on the Micro Channel. The university workload has the lowest Micro Channel utilization, and the very heavy workload has the highest, as expected. In all cases, however, the Micro Channel is not the system bottleneck because it is never utilized more than 10% (figure 4).

Host processor utilization. Figure 5 shows the host processor utilization. Contrary to what some people believe, the host processor is not highly utilized in a network file server environment. On a 50 MHz i486DX processor, the processor is never used more than 30% for all workloads

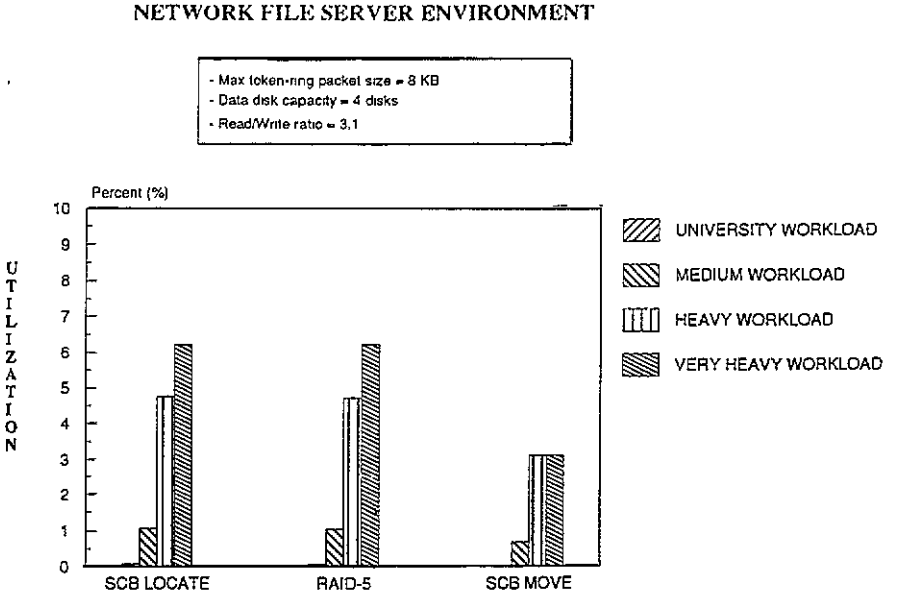

Figure 4. Micro channel utilization.

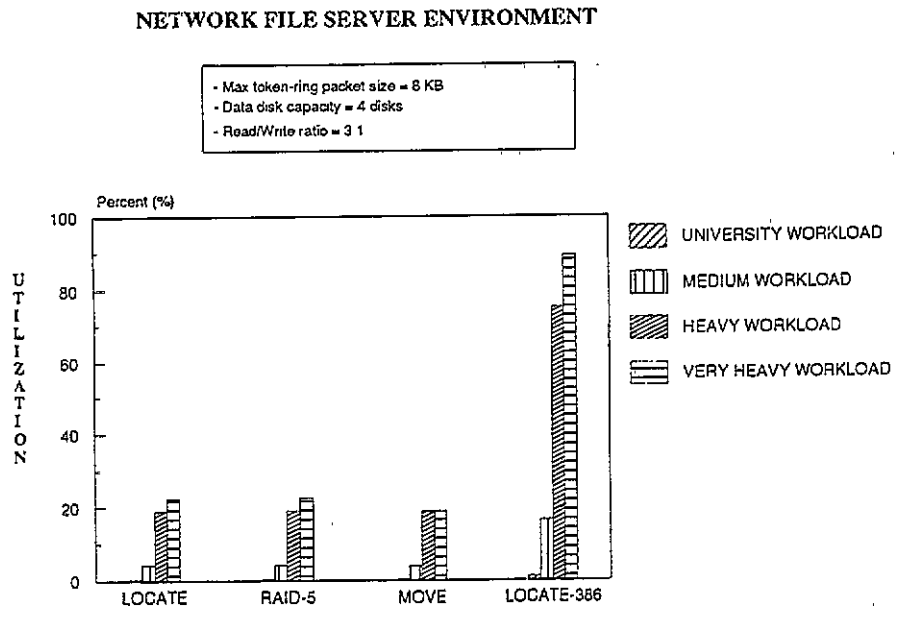

Figure 5. Host processor utilization.

used in this study, so it is not the system bottleneck. If the file server is handling a very heavy workload, an i386-based system may not be adequate, as evidenced by the vertical bars on the right in figure 5, but an i486-based system will do just fine. In the Move Mode, the host processor utilization is not zero in the network file server environment. This is because the host processor, running the operating system's file system and network software, must provide information about each request to one of the two I/O adapters before data can be transferred between those two adapters.

In the Locate Mode and RAID-5 systems, lower file system cache hit ratio means that the host processor has to generate more requests to the disk I/O subsystem and run more file system/disk device driver code per request. Thus the host processor is mot utilized in the very heavy workload, where the file system cache hit ratio is only 20% (figure 5). Since the Move Mode cannot take advantage of the file system cache (or write buffer), the host processor utilization is the same for both heavy and very heavy workloads (Figure 5).

Disk utilization. The data in figure 6 indicates that the average disk utilization in the SCB Move Mode is higher than those in the Locate Mode and RAID-5 systems. This is because the Locate Mode and RAID-5 systems can take advantage of the file system cache and the write buffer in host memory. The cache and write buffer reduce the amount of traffic going to the disk I/O subsystem, resulting in lower disk utilization. As expected, the heavier the

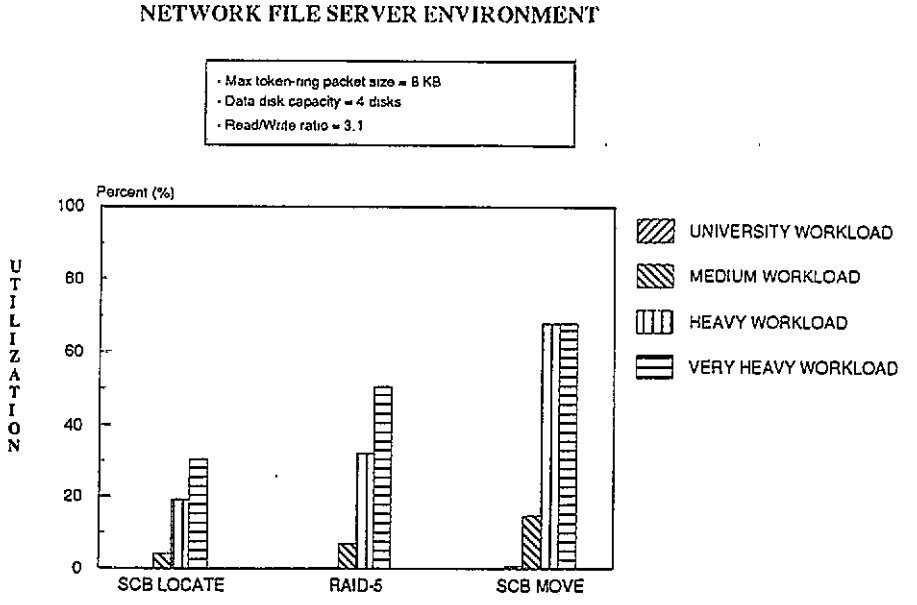

Figure 6. Average disk utilization.

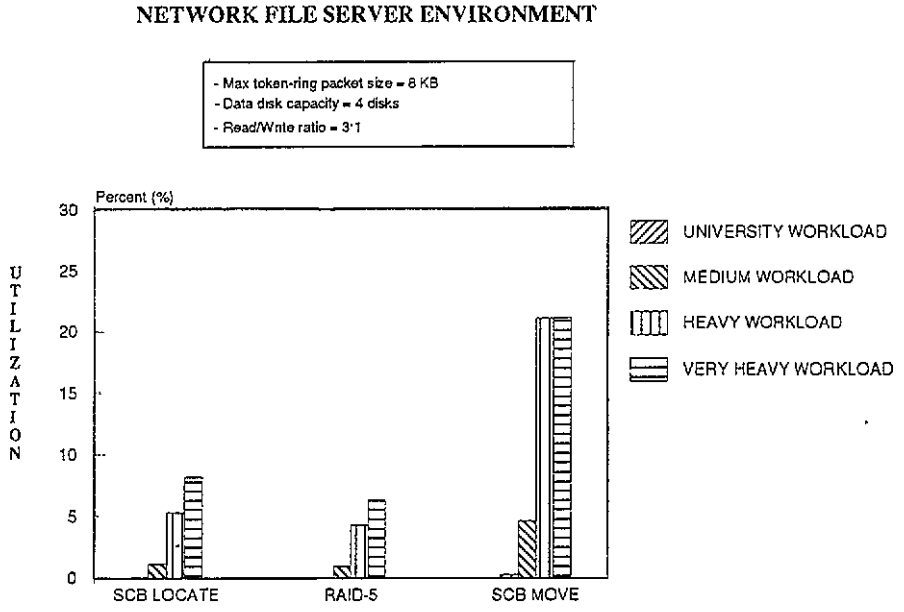

Figure 7. SCSI I/O adapter utilization.

workload, the higher the average disk utilization becomes. Another interesting observation that can be made from figure 6 is that the average disk utilization in the RAID-5 disk array is higher than in a non-RAID disk configuration (Locate Mode). This is because more physical disk accesses are required for a write request or a large read request to the RAID-5 disk array than in the case of non-RAID disk configuration (each disk access requires seek and rotational latencies).

As with other system components, there is no difference in the average disk utilization between the heavy and very heavy workloads in the Move Mode because the Move Mode does not use the file system cache in host memory. From the data in figure 6, the average disk utilization is about 30% in the Locate Mode, over 50% in the RAID-5 array, and close to 70% in the Move Mode, under very heavy workload. Compared with the token-ring utilization in figure 8, the disk utilization in the Locate Mode and RAID-5 systems is still lower than that of the token-ring. However, in the Move Mode under very heavy workload, both the disks and the token-ring can potentially become the system bottleneck.

SCSI I/O adapter utilization. As in the case of disk utilization, the file system cache and the write buffer in host memory help reduce the workload for the disk I/O adapters in the Locate Mode and RAID-5 systems. The Move Mode cannot take advantage of the file system cache and write buffer, so the disk I/O subsystem must handle all traffic. As a result, the SCSI-2 adapter in the Move Mode

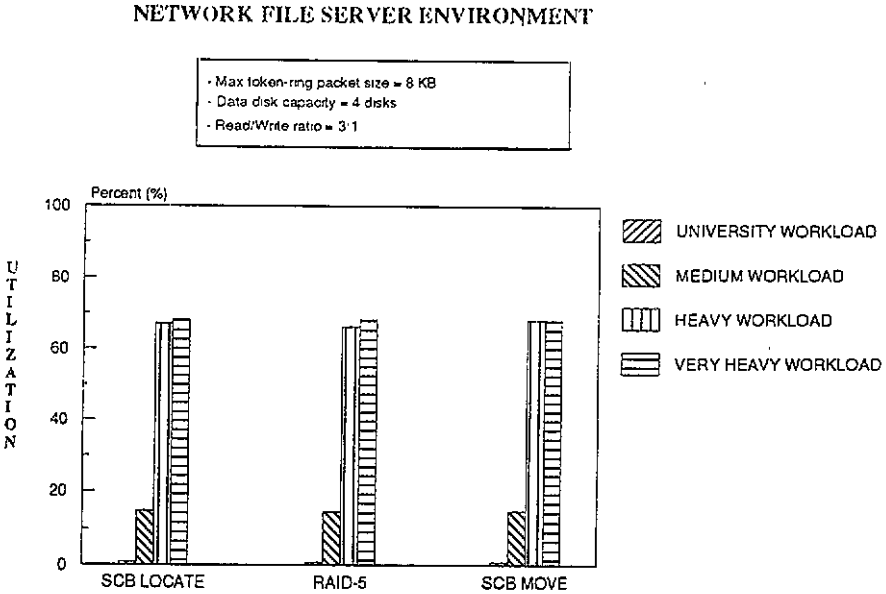

Figure 8. Average token-ring utilization.

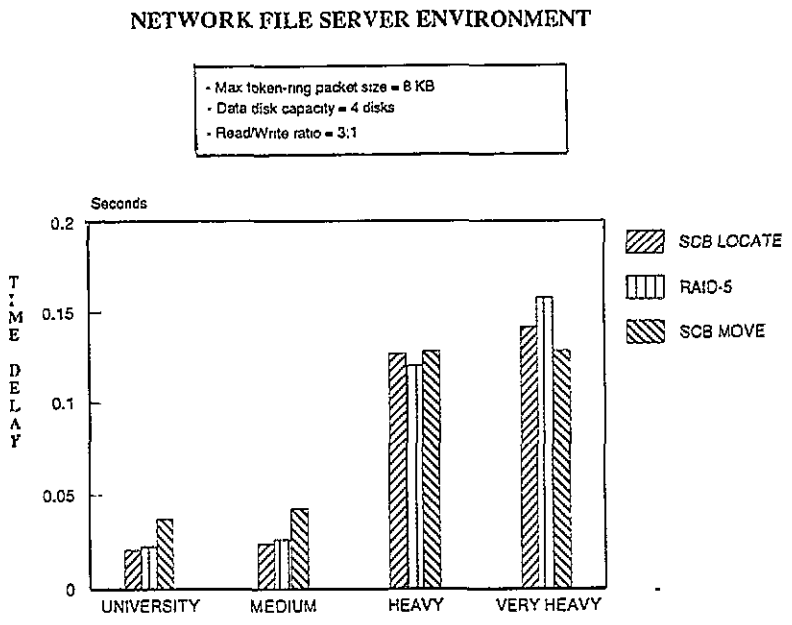

Figure 9. Average overhead per request.

has to process more requests, and becomes more utilized than in the Locate Mode and the RAID-5 adapter (figure 7). The RAID adapter is slightly less utilized than the SCSI 2 adapter in the Locate Mode because the RAID adapter has an i960 RISC processor, which is much more powerful than the 80C186 processor on the SCSI-2 adapter. This is true even though the RAID adapter's i960 processor has to execute more microcode and calculate parity information during disk write requests. However, all disk I/O adapters are less than 25% utilized, so they are not the system bottleneck.

Token-ring utilization. Figure 8 shows that the token ring is utilized more under heavy and very heavy workloads (almost 70% utilized). Thus the token ring can potentially become the system bottleneck under these environments. There is almost no difference in token-ring utilization between the Locate Mode, the RAID-5, and the Move Mode systems, as all three systems have to transmit data on the token ring in the same manner.

Average overhead per I/O request. Figure 9 shows that, the heavier the workload, the higher the average overhead to complete each I/O request because of higher resource utilization and contention. This is quite expected. As mentioned in section 3, the only difference between the heavy and very heavy workloads is the file system cache hit ratio, so the average overhead per I/O request in the Move Mode is the same for these two workloads since the Move Mode does not use the file system cache in host memory.

Under the university and medium workloads, the SCB Move Mode is worse than the Locate Mode and RAID-5

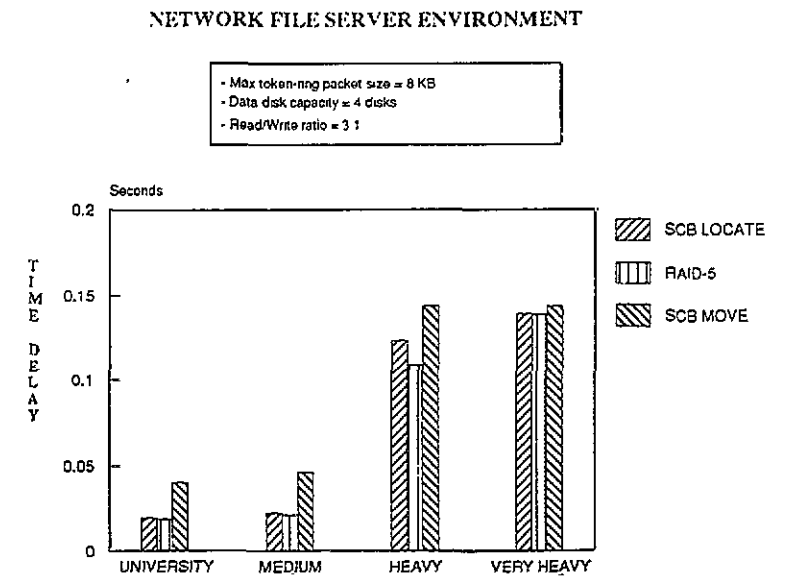

Figure 10. Overhead per read request.

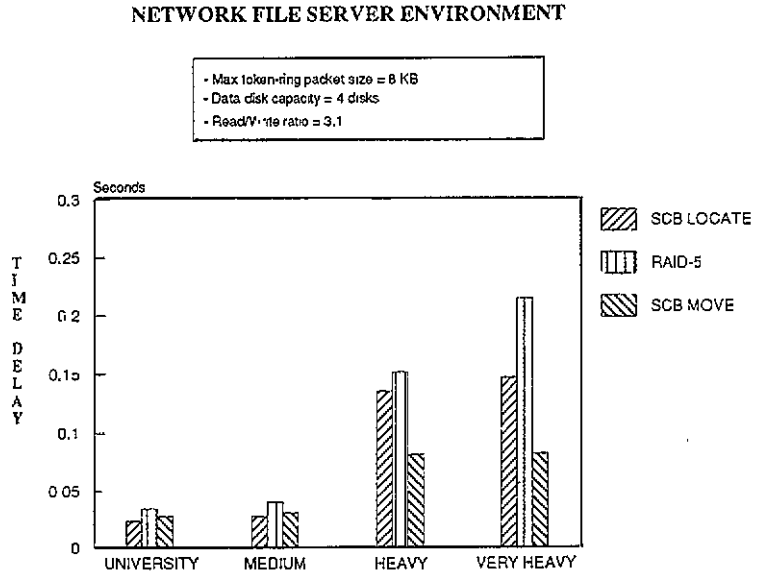

Figure 11. Average overhead per write request.

systems because it bypasses the host memory in peer-to-peer data transfers, and so cannot take advantage of the file system cache and write buffer in host memory. As we have discussed, the file system cache and the write buffer reduce the traffic going to the disk I/O subsystem. This is reflected in figures 10 and 11 as well: for read requests, the SCB Move Mode is the worst among the three systems; for write requests, it is the second-worst. The file system cache can reduce the read traffic going to the disk I/O subsystem by 60% in the Locate Mode and RAID-5 systems. There is not much difference in the write overhead between the Locate Mode and Move Mode because the write buffer used in the Locate Mode can only reduce about 10% of the write traffic. The RAID-5 system is the worst for write requests mainly because of the read-parity-calculation-write operations. In addition, as discussed previously, a write operation requires multiple drives in the disk array to be accessed. Since the disk spindles are not synchronized, there is a performance penalty [10]. However, under the university and medium workloads, the average inter-arrival times of the requests are fairly large, so the Locate Mode and RAID-5 systems are pretty close to one another in performance.

Under the heavy workload, the RAID-5 system is the best overall: it has the lowest average overhead per I/O request. It is followed by the Locate Mode, and then the Move Mode in performance order. For read requests, the spectacular performance of the RAID-5 disk array, as compared to the Locate Mode, is due to several factors. First, the striping of data in 8 Kbyte blocks across all disks allows multiple small requests (8 Kbytes or less) to be processed simultaneously, and large requests to be processed by parallel disks. Secondly, the command queueing capability allowed by the IBM RAID adapter's I/O protocol makes the processing of requests more efficient than the Locate Mode because requests are not queued up in the operating system's device drivers. Thirdly, the adapter's memory buffer cache helps to cut down the read traffic going to the disks by approximately 20%. In lighter workloads, the positive effects of the adapter's command queueing, data striping, and memory buffer cache are not apparent because the requests' arrival rates are relatively large. For write requests, the RAID-5 disk array has the worst performance largely because of the read-parity-calculation-write operations. However, overall, the smaller overhead for read requests more than offsets the larger write overhead in the RAID-5 system (the read-to-write ratio is 3 : 1). Under this heavy workload, the SCB Move Mode—with its peer-to-peer I/O protocol, command queueing, and smaller system resource utilization—approaches the Locate Mode in performance, even though the Move Mode cannot take advantage of the file system cache in host memory for read operations. In fact, the Move Mode already boasts the lowest overhead for write requests, where the write buffer can only reduce the write traffic to the disk subsystem by 10% for the Locate Mode and RAID-5 systems (unlike the 60% reduction for read traffic by the file system cache). This shows how efficient the Move Mode is under heavy workload conditions. The main reason for the Move Mode's efficiency is as follows. Each request involves two slow subsystems, the disk subsystem (which has become much slower in network environments due to higher seek times and the inability to take much advantage of the disks' read-ahead buffers) and the token-ring network. In the Locate Mode, each data transfer must travel from the source subsystem to host memory, and then, from host memory to the destination subsystem; thus, from the standpoint of a single data transfer, the two subsystems operate serially, not in parallel. On the other hand, the Move Mode, with its direct peer-to-peer transfer protocol, allows these two slow subsystems to operate in parallel (one sending and the other one receiving data), and thus, significantly cuts down the overall time delays.

Unsurprisingly, when the file system cache hit ratio drops to 20% in the very heavy workload, the Move Mode

becomes the best system. Its write performance is obviously the best, as in the case of heavy workload discussed above. The write performance for the RAID-5 system is by for the worst because, under very heavy workload, the adapter's memory buffer cache hit ratio for read operations required by parity calculations is much lower than in the case of heavy workload. With the file system cache hit ratio at 20%, the Locate Mode can still benefit from the file system cache, but not as much as in the heavy workload, so with its serial I/O processing nature and higher resource utilization (host processor, Micro Channel, etc), its read performance is comparable with the Move Mode. Consequently, the average overhead per I/O request (including both reads and writes) is the best in the Move Mode, followed by the Locate Mode, and then the RAID-5 system.

6. Conclusion

We have compared the performance of a conventional, interrupt-driven I/O protocol (SCB Locate Mode) with that of a peer-to-peer, shared-memory I/O protocol (SCB Move Mode). We also looked at the performance impact of RAID technology's most popular implementation (a RAID-5 disk array) on a network file server. For the file server environment considered in our study, it appears that the SCB Move Mode achieves the best performance under very heavy, multi-user workload conditions. It also uses less system resources (host processor, memory, Micro Channel)—with the exception of I/O adapters and disks. The exceptions exist because the Move Mode cannot take advantage of the file system cache and write buffer in host memory, and the size of requests must match the maximum network packet size. Under the university and medium workloads, the Locate Mode and RAID-5 systems are superior to the Move Mode because the Move Mode bypasses the host memory in data transfers, and thus, cannot take advantage of the file system cache and write buffer in host memory. Under the heavy workload, the RAID-5 system is the best overall, thanks to its data striping, command queueing, and larger adapter's memory buffer. However, when the file system cache hit ratio drops very low (20%), as in the very heavy workload, the Move Mode post the best performance with its peer-to-peer I/O protocol, asynchronous command processing, and low system resource utilization.

Acknowledgments

The authors wish to thank Dr Neal Coulter of Florida Atlantic University, Dr Dieu Huynh and John Sierra of IBM Corporation for their wonderful support. Their input and constant encouragement have made a significant difference in the progress of this work. Special thanks are also due to Ranga Anumulapally of Florida Atlantic University for his active participation in this project.

References

[1] *Subsystem Control Block (SCB) I/O Architecture Technical Reference* 1991 1st edn (Boca Raton, FL: IBM Corporation)

[2] Bodnarchuk R and Bunt R 1991 A synthetic workload model for a distributed system file server *Proc. 1991 ACM SIGMETRICS Conf. on Measurement and Modeling of Computer Systems (San Diego, CA)* (New York: ACM) pp 57–68

[3] Ganger G and Patt Y 1993 The process-flow model: examining I/O performance from the system's point of view *Proc. 1993 ACM SIGMETRICS Conf. on Measurements and Modeling of Computer Systems (Santa Clara, CA)* (New York: ACM) pp 86–97

[4] Gusella R 1990 A measurement study of diskless workstation traffic on an ethernet *IEEE Trans. Commun.* **38** 1557–68

[5] Jain R and Routhier S 1986 Packet trains—measurements and a new model for computer network traffic *IEEE J. Selected Areas Commun.* **SAC-4** 986–95

[6] Willick D, Eager D and Bunt R 1992 Disk cache replacement policies for network fileservers *Dept. of Computational Science Technical Report* University of Saskatchewan, Canada

[7] Biswas P, Ramakrishnan K and Towsley D 1993 Trace driven analysis of write caching policies for disks *Proc. 1993 ACM SIGMETRICS Conf. on Measurement and Modeling of Computer Systems (CA)* pp 13–23

[8] Bachus K, Houston P and Longsworth E 1993 Right as RAID *Corporate Comput.* 2(5) 61–5

[9] Patterson D, Chen P, Gibson G and Katz R 1989 Introduction to redundant arrays of inexpensive disks *Proc. IEEE COMPCON* (Los Alamitos, CA: IEEE) pp 112–7

[10] Ng S 1989 Some design issues of disk arrays *Proc. IEEE COMPCON* (Los Alamitos, CA: IEEE) pp 137–42

[11] MacNair E and Sauer C 1985 *Elements of Practical Performance Modeling* Englewood Cliffs, NJ: Prentice Hall pp 46–80