

Using the SAS® System under AIX® Effectively

Maria L. Moore, SAS Institute Inc., Austin, TX

ABSTRACT

This paper discusses issues involved in tuning the performance of the SAS System under the AIX operating system. In order to optimize performance, both the allocation of hardware/software resources and the setting of SAS system options must be considered. Included are discussions on SAS system options, hardware configurations, I/O constraints, disk usage, memory size and usage, and network traffic.

INTRODUCTION

IBM® developed the second generation Reduced Instruction Set Computer (RISC) architecture family with the goals of satisfying customer requirements for both scientific and commercial applications and providing the durability necessary for extensions through several technology generations. Their efforts culminated in the release of the RISC System/6000™ family. The family consists of the Xstation 120, POWERstation 320, POWERstation 520, POWERstation 530, POWERserver 540, POWERstation 730, and POWERserver 930.

The SAS System under the AIX operating system was released for the IBM RISC System/6000 machines in February 1991. The SAS System requires AIX Release 3.1 or later, 8M (megabytes) of memory, and 100M of fixed disk for the entire SAS System and all optional files.

The performance of the SAS System depends on the

machine size, the amount of internal memory available, the disk configuration, the SAS system options in effect, and the number of SAS sessions running, as well as the network traffic when accessing files over a network.

This paper does not attempt a comprehensive coverage of all topics but will touch on each area and point out valuable hints for users. Research is ongoing in all these areas and new findings will be published as they become available.

TESTING ENVIRONMENT

All performance tests were run on the RISC System/6000 machines listed in Table 1. The desktop model 320 unit was defined for entry level requirements of CPU, memory, and disk storage. The desktide model 520 contains greater memory, disk storage, and expansion capabilities. The desktide model 530 provides the greatest CPU power and storage of the three machines tested.

One version of the SAS System was physically located on the fixed disks of the RISC System/6000 Model 530 on logical volume hd2 for local testing. Another version of the SAS System was physically located on a RISC System/6000 model 520 on logical volume lv00, then mounted on the Model 530 using the Network File System (NFS) via Ethernet. Table 2 shows that logical volumes can be split among physical disks.

Model	Clock Speed	MIPS	Dhrystones K/sec	Linpack DP MFLOP	Data Cache	Instr Cache	Memory Bus	Memory	Disk Space
320	20MHz	27.5	48.3	7.4	32K	8K	64b	16M	640M
520	20MHz	27.5	48.3	7.4	32K	8K	64b	40M	2.3G
530	25MHz	34.5	60.7	10.9	64K	8K	128b	48M	2.3G

Table 1: Comparison of RISC System/6000 Systems

Model	Disk Space	Disk Configuration Logical Volumes
320	640M	hdisk0 320M - 32M pagespace /tmp
520	2.3G	hdisk1 320M - 32M pagespace hdisk0 670M - 64M pagespace hd3(/tmp)
		hdisk1 670M - lv00 hdisk2 355M - 96M pagespace lv00
530	2.3G	hdisk3 670M - lv00 hdisk0 670M - 104M pagespace hd2, hd3(/tmp)
		hdisk1 670M - lv00 hdisk2 355M - 48M pagespace hd2, hd3(/tmp)
		hdisk3 670M - lv00, lv02

Table 2: Disk Configuration

Two SAS jobs were used to run the tests:

- A CPU-intensive SAS job consisted of sorting 100,000 observations of ten numeric variables with five sort keys. The sort required 9.08M of memory.
- An I/O-intensive SAS job consisted of reading and writing sets of 100,000 observations of length: 20 bytes, 40 bytes, 60 bytes, 80 bytes, 120 bytes, 160 bytes, 200 bytes, 300 bytes, 400 bytes, and 600 bytes each.

SAS/GRAPH® software was used to graph test results.

COLLECTING PERFORMANCE STATISTICS

Before attempting to improve performance it is important to know what the SAS job is doing, how it is doing it, and where the bottlenecks occur. For example, it does not help to increase the I/O speed if the job is CPU bound. Under many circumstances, reducing the use of any of the critical computer resources, I/O, memory, or CPU time results in better throughput of a particular job.

The first step in improving the performance of a SAS job is to write efficient SAS code. A good reference is *SAS Programming Tips, A Guide to Efficient SAS Processing*. Chapter 2 covers performance statistics, SAS data set information, and verifying programming techniques. Other chapters provide useful information for writing efficient SAS code. Another good reference is Chapter 12, "Optimizing System Performance," in *SAS Companion for the UNIX Environment and Derivatives, Version 6, First Edition*.

After tuning the SAS code to be as efficient as possible, examine the hardware configuration to determine how to further increase the performance. The AIX operating system supplies several standard UNIX commands to aid in determining problem areas. These include `ps`, `netstat`, `lstat`, and `vmstat`.

Figures 1 and 2 show the output from `lstat`. Figure 1 shows that disk0 was active approximately 30 percent of the time during the I/O-intensive SAS job. Figure 2 shows that all the disks were active no more than 4 percent during the CPU-intensive SAS job. The small amount of activity for disk0 was due to paging. These

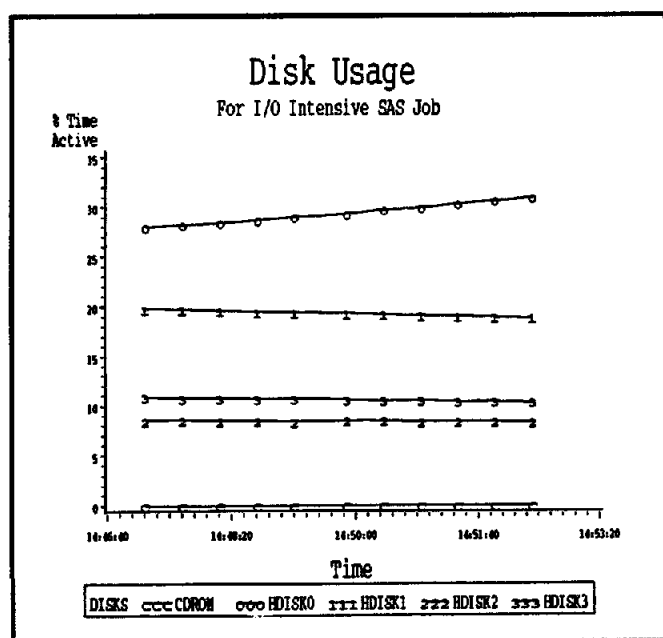


Figure 1

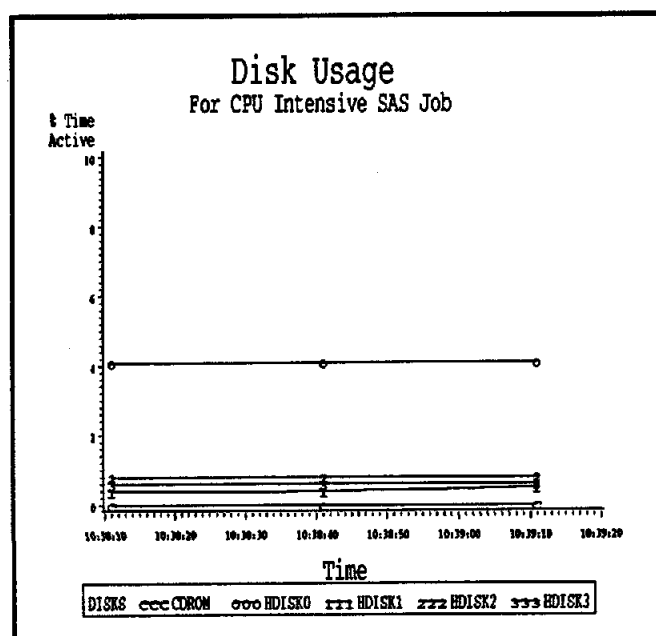


Figure 2

graphs show whether improving I/O performance will affect overall performance.

Figures 3 and 4 show the output from ps. Figure 3 shows that CPU usage peaked at 60 percent during initial startup, but then dropped to under 40 percent during the I/O-intensive job. Paging continued to increase as the size of the I/O increased. Compare that to Figure 4, where paging remained low while CPU usage increased to close to 100 percent when the CPU-intensive job was executed. These graphs show whether improving CPU performance will affect overall performance. By studying similar charts for particular SAS jobs, areas of high usage can be identified and targeted to increase performance.

MACHINE SIZE

Although it is not always practical, the performance of a SAS job can be increased by running on a faster model machine. The clock speed of the three machines tested was very close. Since the Model 320 had only 16M of memory, the test cases for this comparison were run on the Model 520 and Model 530 with a simulated memory size of 16M. Therefore, any performance increases due to the additional internal memory would not be seen.

Times shown in Figures 5 and 6 were collected using the FULLTIMER system option. Real time represents the actual clock time it took to execute the job. System CPU time is processor time spent executing kernel calls and system routines. User CPU time is CPU time spent executing the SAS code.

The tests revealed that there is not much difference between these three model machines. The system real time and user real time show slight gains as you move to a faster machine. However, throughput, as measured by real time, is much more dependent on disk resources and file system allocation than on machine speed. As Figures 5 and 6 show, no matter what model machine is used, throughput can suffer.

INTERNAL MEMORY

A minimum of 8M of memory is suggested for running SAS software. By increasing the amount of physical memory, you can decrease the processing time because the amount of time spent on paging is reduced.

The test machine had 48M of internal memory. For testing purposes a tool was utilized which simulated an RISC System/6000 workstation with various sizes of real memory. The driver program executed a SAS job twice (1 warmup + 1 measured) over a range of memory sizes, collecting performance statistics for each memory size.

The program was run using the I/O-intensive SAS job and the CPU-intensive SAS job. Each job was run once writing to the same disk where the system page space was allocated and once writing to a separate disk.

The I/O-intensive job, shown in Figure 7, shows the greatest decrease in real time between 8M and 16M, with smaller decreases following for each additional 8M of added memory.

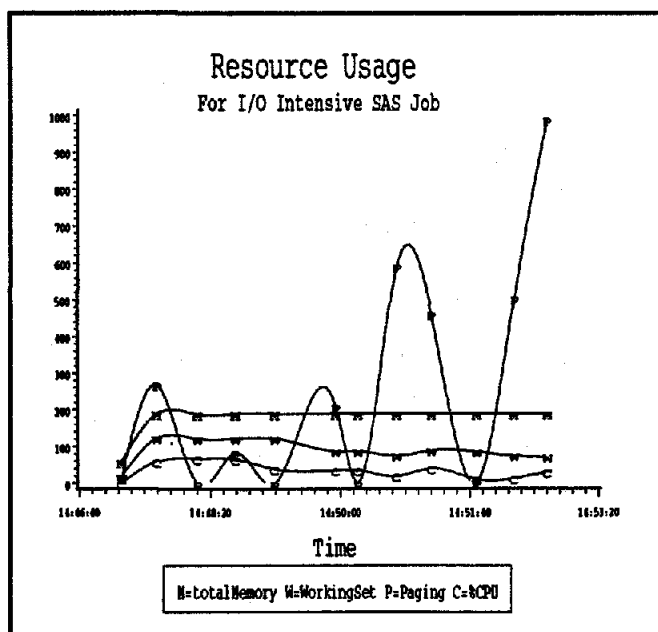


Figure 3

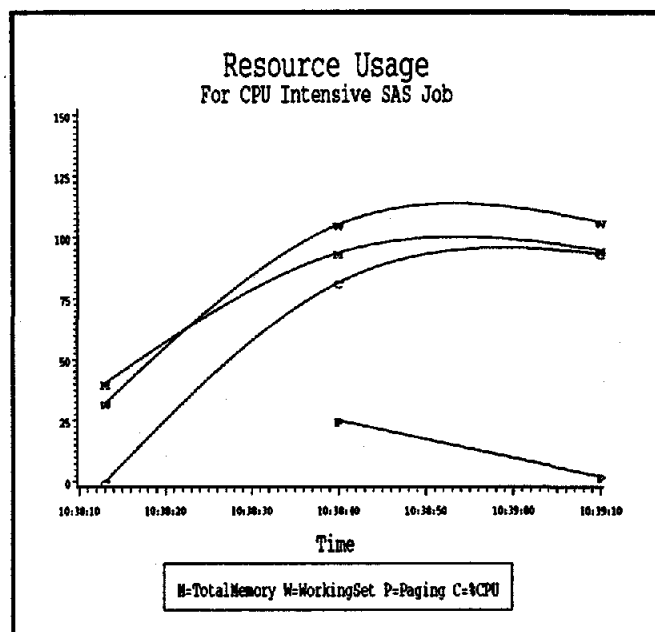


Figure 4

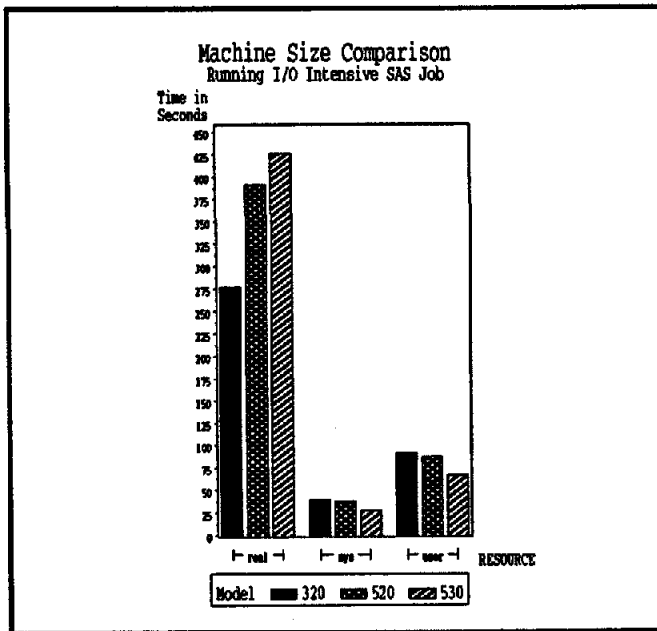


Figure 5

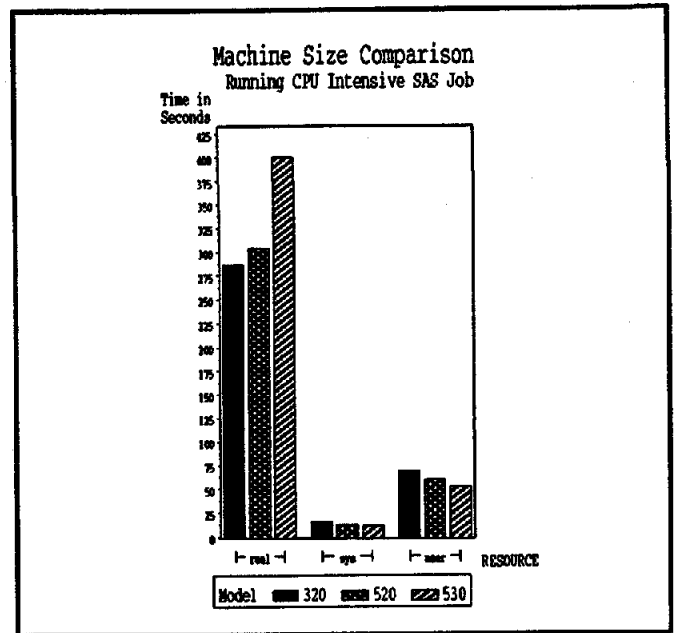


Figure 6

Figure 8 shows that gains continue when up to 32M of memory is added for CPU-intensive jobs. A machine with 8M of internal memory was determined to be impractical to run CPU-intensive jobs, which took well over 7 hours to complete. The paging rate, shown in Figures 9 and 10, drops for each additional 8M of memory.

Figures 7-10 also show that the performance can be improved by writing to a separate disk from the one in which the system page space is allocated. By reducing disk drive contention the paging rate is increased, thereby

lowering real time and increasing performance.

DISK DRIVES

Overall performance depends on how file systems and page space are allocated on the fixed disks and where the SASWORK directory is temporarily stored. System CPU and user CPU remain relatively constant regardless of where the files reside. However, real time can vary greatly.

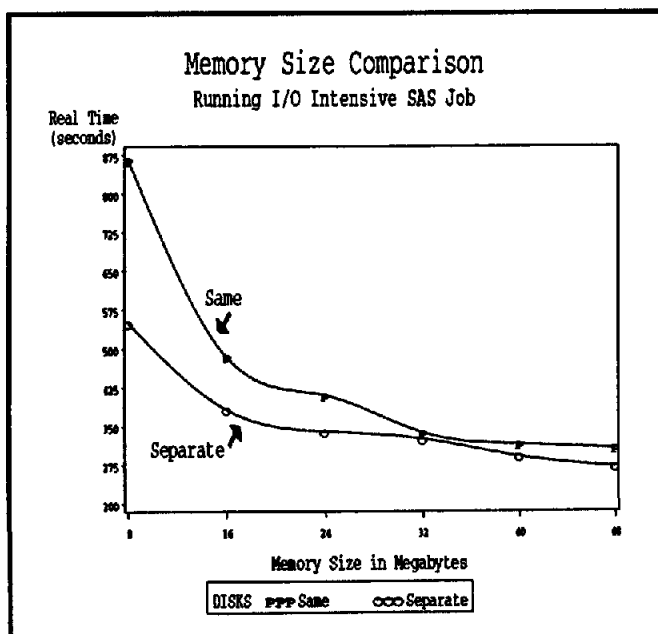


Figure 7

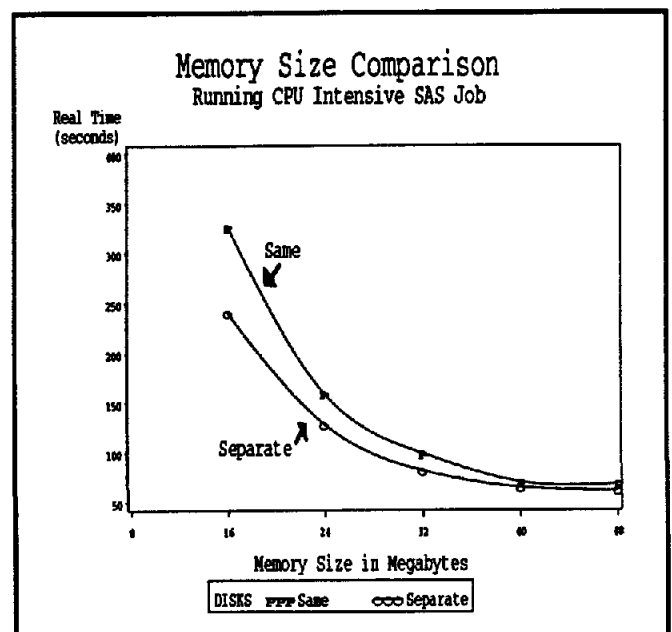


Figure 8

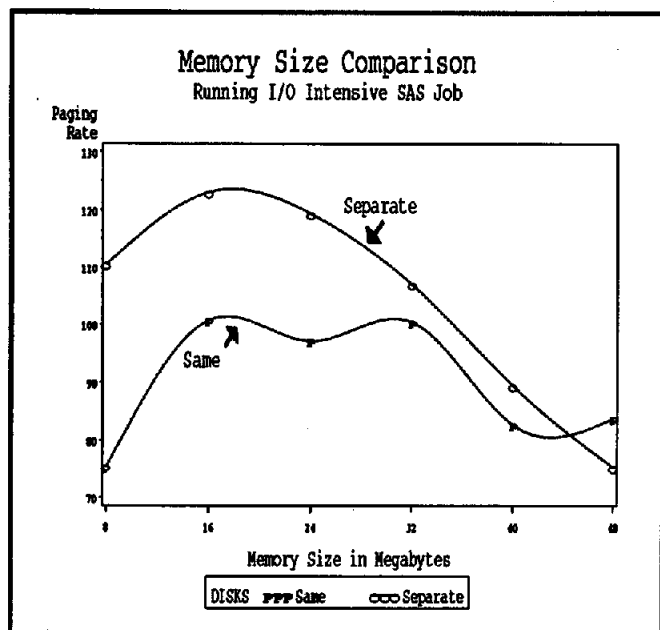


Figure 9

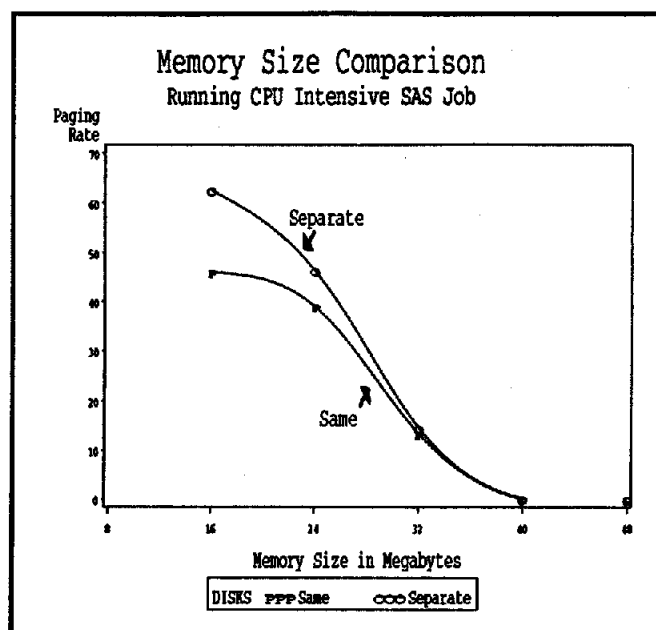


Figure 10

The goal is to minimize system wait time caused by disk contention, thereby increasing throughput. Therefore, it is necessary to know where the system page space is allocated, where the SAS System is stored, where the SASWORK directory is stored, and where any permanent SAS directories are stored.

By default, the SASWORK directory is stored in /tmp. The simplest way to improve overall throughput is by carefully selecting where the SASWORK directory is stored using the WORK SAS system option.

Five SAS jobs were executed concurrently during three separate sessions where the disk assignment of the SASWORK directory of each job was varied between sessions using the WORK option. This was repeated for both the I/O-intensive SAS job and the CPU-intensive SAS job.

Figure 11 shows the number of SAS processes using each physical disk during each session. Session 1 had disk accesses evenly distributed across all four disks. Session 2 had a few more processes using disk 3. Session 3 had all the processes using disks 1 and 3.

To achieve the greatest throughput, disks' accesses must be evenly distributed across physical disks for all sessions that are running. Notice that the WORK option does not actually point to a physical disk, but to a logical volume, which may be split among physical disks.

Figures 12 and 13 show that moving the SASWORK directory from one physical disk to another can greatly change the throughput of an individual SAS job as well as

any other SAS jobs being run simultaneously. The two figures show the effects of moving the SASWORK directory with multiple users running I/O-intensive jobs or CPU-intensive jobs.

During Session 1 of the I/O-intensive job, throughput was approximately equal for Users 3, 4, and 5 since disk contention was kept to a minimum. However, Users 1 and 2 were accessing disks with higher activity rates and experienced longer real times. During a CPU-intensive job, User 3 experienced slowed throughput compared to Users 4 and 5, since User 3 was writing to the same disk

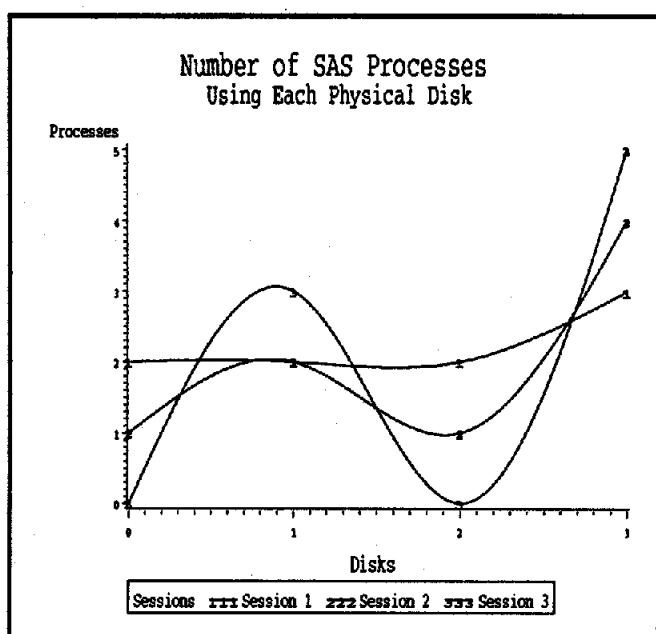


Figure 11

where system page space was allocated.

During Session 2 of the I/O-intensive job (Figure 12), User 3 was not competing for the disks and experienced the fastest throughput. Users 1, 2, 4, and 5 were all accessing the same disks, which slowed throughput for them all. Similar results were seen when testing the CPU-intensive job (Figure 13).

During Session 3, all five users accessed the same two disks. Users 4 and 5 experienced faster throughput when they ran I/O-intensive jobs but not when they ran CPU-intensive jobs.

Table 3 shows the average real time for each session. For the I/O-intensive job, Session 1 ran an average of 23 percent faster than Session 3. There was an even greater increase when running the CPU-intensive job, where Session 1 ran an average of 79 percent faster than Session 3.

MULTIPLE SAS SESSIONS

Multiple sessions of the SAS System can be executed by multiple users on a single workstation. Rarely does one process have 100 percent of the machine resources to itself. As more processes share a particular resource, less of that resource is allocated to each process.

The number of concurrent SAS sessions running affects the performance of each session. System CPU and user CPU remain relatively constant regardless of the number of concurrent SAS sessions running. However, real time can vary greatly.

To test the effect of concurrent sessions on real time, the I/O-intensive SAS job was run by one user with the FULLSTIMER option specified. Additional users were added, one at a time, each running the same job with the FULLSTIMER option specified. Two disks were used, each new user writing to one or the other disk. The same sequence was followed with a CPU-intensive SAS job.

Figure 14 shows that when two users were running, writing to different logical volumes, both had about the same real time, which was about 30 percent slower than just one user running the same job. When User 3 is introduced he competes for disk resources with User 1, which slows both of them down but does not impact User 2 at all. User 4 competes with User 2, which slows both of them down. The average real time for five sessions running concurrently was 1469.24 seconds.

The average real time was improved considerably by increasing the number of disks. The same I/O-intensive SAS job was run by six users writing to four disks evenly

	I/O Intensive	CPU Intensive
Session1	1198.48	670.898
Session2	1310.82	970.944
Session3	1469.24	1202.958

Table 3: Average Real Time in Seconds

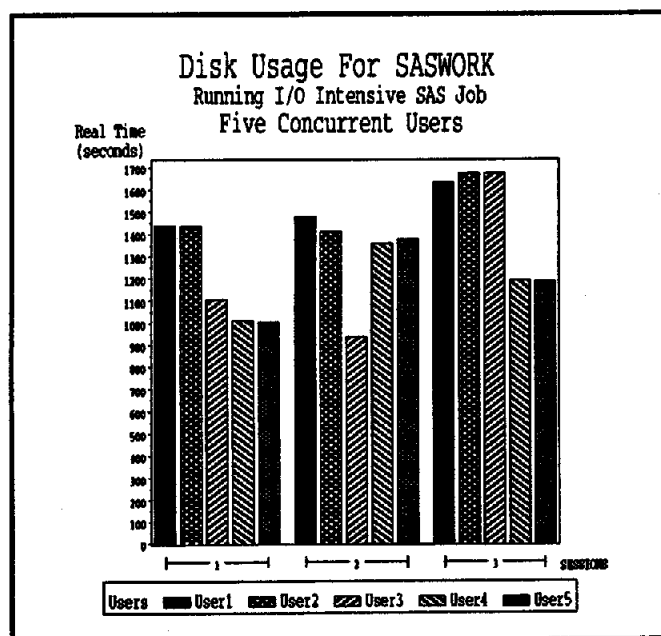


Figure 12

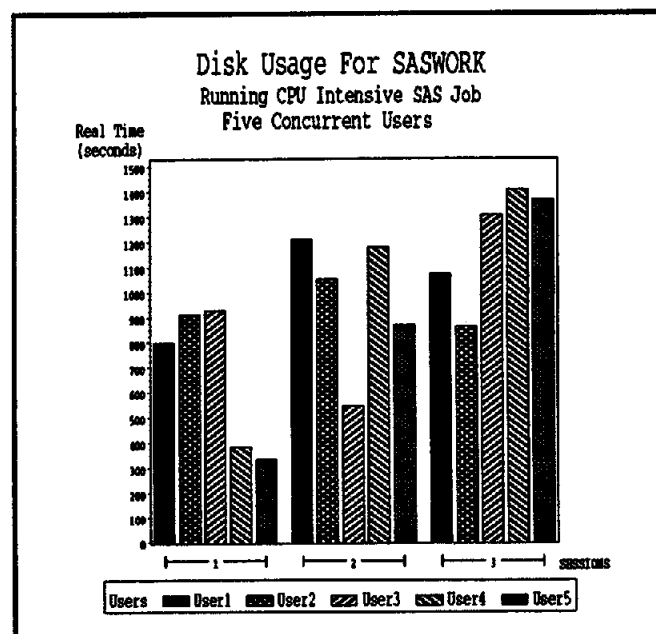


Figure 13

distributed among them. The resulting average real time was 1394.9 seconds, a gain of 5 percent over the five users on two disks. For the CPU-intensive SAS job, the respective averages were 1202.96 seconds and 815.73 seconds, a gain of 47 percent.

This clearly shows that it pays to coordinate disk usage among both users and sessions. Note, however, that when five or more SAS sessions were running concurrently, interactive response time was severely degraded.

NETWORKS

The SAS System can be used in a network environment. Workstations may be connected via NFS mounts so they share a file system.

To test network performance, multiple SAS sessions were run with the SAS System residing on a local machine. Next, multiple SAS sessions were run with the SAS System residing on a remote machine with the SAS files being loaded with NFS over Ethernet. In both cases the SASWORK directories were stored on the local machines.

The average real time for both I/O-intensive and CPU-intensive SAS jobs was very close when the SAS System was loaded using NFS as well as when it was loaded locally. In some instances there was a slight improvement using NFS, since disk contention on the local disks was decreased.

Figure 15 shows that as additional sessions are run, the

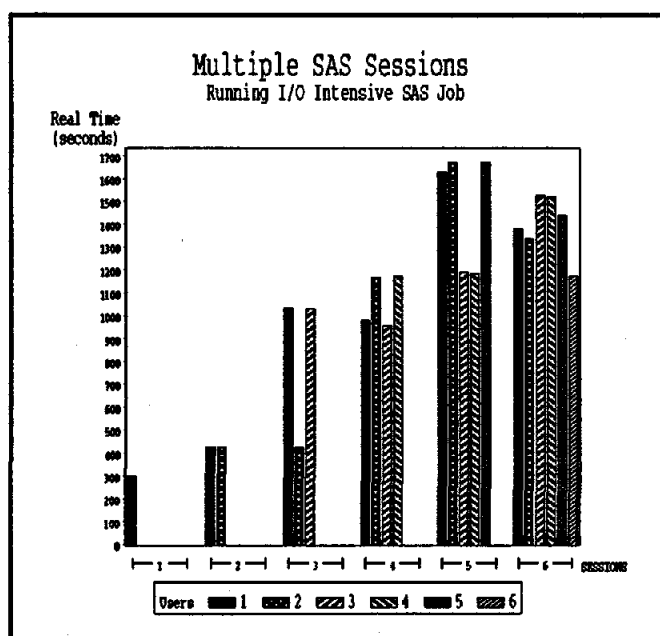


Figure 14

average total CPU time for local processing remains relatively constant. However, when NFS is used for loading the SAS System, many external factors come into play. Tests showed that for each user added, system wait time due to a NFS read increased by 25 percent for I/O-intensive jobs and was slightly higher for CPU-intensive jobs. However, Figure 15 shows that as additional sessions are executed the factors tend to balance themselves.

SAS SYSTEM OPTIONS

Significant performance gains can be achieved by using the SAS system options wisely. One example of this has already been shown using the WORK option.

The SORTSIZE option limits the amount of memory available to the SORT procedure. The default is set to 8M. By adjusting the SORTSIZE, the amount of swapping the SAS System must do to sort a data set may be reduced. *SAS Companion for the UNIX Environment and Derivatives* states that the SORT procedure's algorithm can swap unneeded data more efficiently than the operating system can. It advises that if the machine has more than 6M of physical memory and the data set being sorted is large, setting the SORTSIZE option to a value greater than 2M may improve performance.

The optimal SORTSIZE is dependent on internal memory, how large the data set to be sorted is, and the machine load.

The testing program sorted 100,000 observations of ten numeric variables with five sort keys and used 9.08M of memory. The test was run across various memory sizes using various SORTSIZE options.

Figure 16 shows that with internal memory of 16M or 24M, a sortsize of 2M results with better throughput. However, with 32M or more of internal memory, sortsizes of 2, 4, 6, and 8 all have very similar results.

When sorting large data sets on a machine with a small amount of internal memory, there is a tradeoff. If the sortsize is small, leaving internal memory enough for loading of the SAS System, the paging rate increases for the data. If the sortsize is large, giving plenty of room for the data, the paging rate increases for the SAS System. The goal is to find the sortsize which minimizes the paging rate for both the data and the program, since real time goes down as the paging rate goes down.

SUMMARY

The I/O-intensive SAS job and the CPU-intensive SAS job were run on the Model 530, with 48M of internal memory and using the SAS system defaults: loading the SAS System from the local machine, SORTSIZE option of 8M, and storing the SASWORK directory in /tmp which is on the same physical disk as the system page space. Those same two SAS jobs were run again, optimizing the overall throughput using the techniques which have been discussed. The results are shown in Table 4.

The real time for the I/O-intensive SAS job was improved 1 percent by loading the SAS System using NFS and storing the SASWORK directory on a disk that does not contain the system page space.

The real time for the CPU-intensive SAS job was improved 21 percent by loading the SAS System from the local machine, using a SORTSIZE option of 4M, and storing the SASWORK directory on a disk that does not contain the system page space.

Not all SAS jobs are created equal. What works for increasing the performance of one, may in fact, degrade the performance of another. It pays to know your SAS jobs, your machine, and your environment.

FUTURE CONSIDERATIONS

In the future, testing will continue in the following areas:

- Running the SAS System from an Xstation
- Increasing system page space
- Running multiple sorts while varying the sortsize

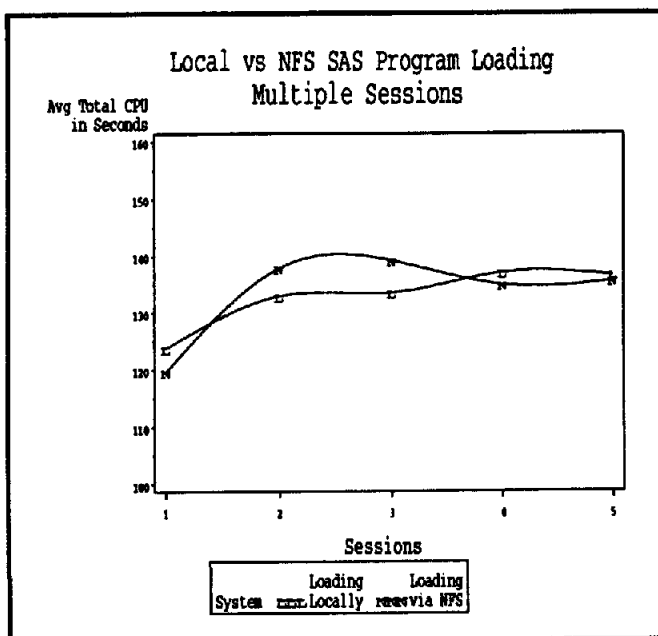


Figure 15

	Defaults	Optimized
I/O intensive:		
Real time	186.742	184.132
User CPU	65.720	70.440
System CPU	22.260	24.260
CPU intensive:		
Real time	72.314	59.707
User CPU	51.900	49.280
System CPU	8.050	8.280

Table 4: Defaults vs Optimized SAS Jobs

- Ethernet versus token ring
- Writing to SASWORK directories using NFS
- SAS system options:
 - BUFNO
 - BUFSIZE
 - SHAREBUFFERS
 - SYSLEAVE
 - PROCLEAVE

SAS and SAS/GRAPH are registered trademarks of SAS Institute Inc., Cary, NC, USA.

IBM and AIX are registered trademarks of International Business Machines Corporation.

RISC System/6000 is a trademark of International Business Machines Corporation.

Other brand and product names are registered trademarks or trademarks of their respective companies.

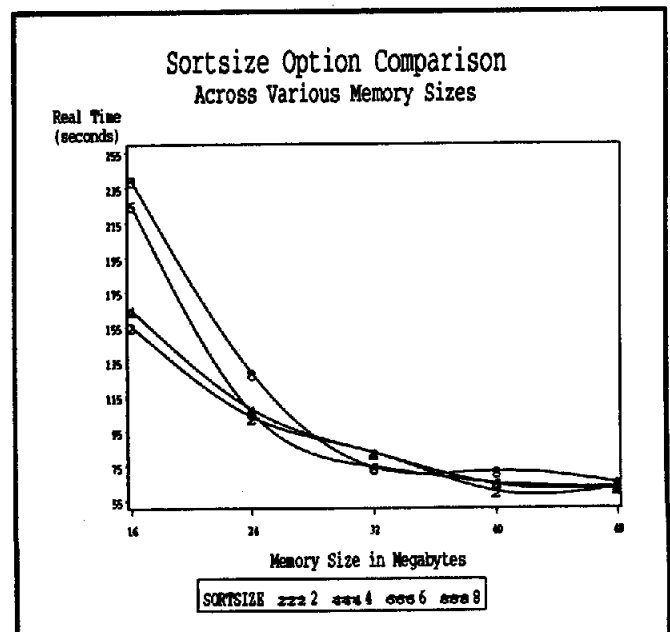


Figure 16