

Maximizing SAS® Software Performance Under the Unix Operating System

Daniel McLaren, Henry Ford Health System, Detroit, MI
George W. Divine, Henry Ford Health System, Detroit, MI

Abstract

The Unix operating system has gained widespread acceptance in recent years. Unix provides a fast, secure, and reliable platform for running the SAS System. This paper will describe how to maximize SAS software performance on your Unix system using techniques that apply to most variants of the Unix operating system.

Topics covered include performance benchmarks, configuring memory, swap space and *tmpfs*, maximizing disk subsystem performance, enhancing multiuser performance, reducing disk space requirements, setting SAS system options, setting process priorities, eliminating unnecessary system processes, and monitoring performance.

Background

The Henry Ford Health System Department of Biostatistics and Research Epidemiology (BRE) currently runs Release 6.12 of the SAS System on a Sun Microsystems SPARCserver 670MP running Solaris 1.0 (SunOS 4.1.3). The system was installed in 1992 to replace a Sun 3/280 system that was severely overloaded.

The system was initially configured with 128MB of RAM, 10GB of disk space, and four processors. The system has been upgraded over the years and now has 192MB of RAM and 16GB of disk space.

The system comfortably supports up to 15 simultaneous SAS software users, several users of an Ingres database application, and occasional use of other statistical packages.

The system is configured as a timesharing server, and the SAS System is run interactively from personal computers via an enhanced Telnet terminal emulation package. The performance tips described in this document should apply to other systems configured as timesharing servers as well as those that are configured as remote application servers using SAS/CONNECT® or SAS/SHARE® software.

Performance Benchmarks

The use of a benchmark program (or a suite of benchmark programs) is critical when comparing performance between systems and in measuring the effects of system changes over time. The selection of a benchmark program can be difficult. The benchmark that you select should represent the kind of workload that your system will be expected to support. Our benchmark consists of a single SAS program that we refer to as *kappa*.

Early in 1992, a BRE statistician was trying vainly to find sufficient computing resources to execute a simulation program in SAS. The program was designed to test for a difference in paired kappa statistics by the use of resampling. The generation of the resampling distribution used a great deal of CPU power, and the creation of many temporary data sets also caused the program to be very disk intensive.

The *kappa* program was first run on an IBM® mainframe under the TSO® operating system. The program was never allowed to run to completion on that system since the system operators terminated it whenever it ran for 24 hours or more.

The *kappa* program then was downloaded to our Sun 3/280 system and executed there. The program ran in four hours and 21 minutes on the 3/280.

When we began searching for a replacement for the 3/280 in 1992, the *kappa* program was selected to be our benchmark for system testing since we wanted to have sufficient resources on hand in the future to run *kappa* (and similar jobs) with a much shorter turnaround time. Also, our experience monitoring and documenting the performance of the *kappa* program gave us some firm numbers to work with. The program was also well-suited for benchmarking since it exercised both the CPU and the disk subsystem of the systems under test.

We ran the *kappa* benchmark on various Unix systems supplied by Sun Microsystems, IBM, Hewlett-Packard and Digital Equipment Corporation. The Sun SPARCserver 670MP system was chosen to replace our 3/280 system after comparing its performance to the other systems. The 670MP was capable of running the *kappa* program in as little as 47 minutes, depending on how the system was configured. This was not the fastest time recorded (a system from HP was actually slightly faster), but other factors, including multiuser performance (described below), weighed in favor of the 670MP.

Configuring Memory, Swap Space and *tmpfs*

In general, memory has a larger effect on system performance than any other factor. Insufficient memory will always result in sluggish performance, regardless of the speed of a systems processors, disks, network interfaces, etc.

The SAS Institute recommends configuring your system with 24MB of RAM, plus 8MB for each additional SAS Software user on a

Unix system. We recommend starting with 64MB of RAM, even for a single user, since you will need additional memory for the operating system, your windowing system, and any other applications that you want to use in addition to SAS.

Disk caching, or storing all or part of recently accessed files in memory, is one way of using memory to improve system performance. Disk caching is performed automatically by most variants of the Unix operating system. SunOS 4.1.3, for example, uses any memory not already allocated to running processes to disk caching, so installing additional memory into a system with insufficient memory will not only increase memory performance, but disk performance as well.

Under SunOS, the *tmpfs* filesystem is yet another way to use memory to improve system performance. The *tmpfs* filesystem is essentially a RAM disk which can be used to store temporary files in memory. Files stored on a *tmpfs* filesystem are stored in RAM (or system swap space), rather than written to disk, resulting in greatly improved I/O performance.

Although the *tmpfs* filesystem can greatly increase SAS software performance, that increase in performance does not come without cost. As its name implies, *tmpfs* is a temporary filesystem. When your Unix system is shutdown (or if it crashes), all information in the *tmpfs* filesystem is lost.

Also, since the *tmpfs* files are stored in RAM, they use up memory that would normally be available for executing programs. If the *tmpfs* fills up, it can cause problems for all processes that are executing on your system. To prevent this, we recommend running a script every 10 minutes throughout the day to monitor the *tmpfs* filesystem and to notify the system administrator if it is becoming full (see Listing 1).

The usual cause of a full *tmpfs* filesystem is SAS work files left behind after the abnormal termination of a SAS software program.

```
#!/bin/sh
#
# Script: watch_disks
#
# Purpose: Runs every 10 minutes to look for full
# partitions. Emails "admin" group if any partition
# reaches 100%, or if the swap partition reaches
# 80% of capacity.
#

host=`hostname`

while [ 1 = 1 ]
do
    disk=`df | grep dev | grep "100%" | grep -v
    sr0 | wc -l`

    if [ $disk -gt 0 ]
    then
        df | grep dev | grep "100%" | grep -v sr0 |
        /usr/ucb/mail -s "Warning - Full Filesystem
        on $host" admin
    fi

    swap=`df | grep swap | awk ' { temp = $3 / $2 }
    END { printf ("%d",temp) }'`

    if [ $swap -gt 80 ]
    then
        df | grep swap | /usr/ucb/mail -s "Warning -
        High Swap ($swap) Usage on $host" admin
    fi

    sleep 600
done
```

Listing 1

Until recently, our system was configured to mount a filesystem called */tmp/saswork* at boot time as a *tmpfs* filesystem. All SAS work files were then stored in this filesystem. In our experience, SAS Software performance for disk intensive jobs is doubled when using the *tmpfs* filesystem. SAS work files are well-suited for storage on a *tmpfs* filesystem since they are both created and removed during processing and their loss in a system crash is of no consequence.

Maximizing Disk Subsystem Performance

SAS software is often (wrongly) seen by system administrators as a "performance pig," especially in terms of its I/O requirements. SAS software itself is not the problem - any I/O problems stem from the size of the data sets that are processed. It is not unusual for a BRE statistician to process data sets of 500MB or more. Since a portion of the work that is done is exploratory in nature, it is also not unusual for a statistician to process the same data set repeatedly throughout the day. This type of activity will cause anyone monitoring system performance to sit up and take notice.

There is, however, much that can be done to reduce the impact of processing these large data sets on a given system. First of all, you must have sufficient memory, as described above. A shortage of memory will result in a system that becomes noticeably sluggish due to the swapping of the contents of memory to disk.

Next, make sure that your system has plenty of spare disk space available. SAS software programs that process large data sets will require large amounts of temporary work space in the SAS work library. Programs that sort large data sets will need space for temporary sort files in the directory containing the data sets. Programs that require large amounts of memory will also require large amounts of swap space.

When configuring your disk subsystem, you will want to carefully plan the layout of your filesystems. Make sure that you have plenty of swap space, particularly if you choose to use the *tmpfs* filesystem described above. For maximum performance, configure swap partitions on more than one physical disk. If disk I/O becomes a performance bottleneck, consider adding additional disks, or disk controllers, to split the I/O load across as

many physical disks as possible.

If your system hardware and/or software provides support for RAID (Redundant Array of Inexpensive Disks), your disk performance can be substantially improved by using such techniques as disk striping.

Enhancing Multiuser Performance

If you expect to have many simultaneous SAS software users on a single machine, we recommend using a multiprocessor system. A well designed multiprocessor system will support many simultaneous users without noticeable degradation in performance. In our experience it is possible for a multiprocessor system to simultaneously support several interactive SAS software sessions as well as several compute-intensive batch jobs and still be responsive.

When we were benchmarking systems in 1992, we simulated multiuser loads on each system by running two simultaneous *kappa* benchmarks. On all but one of the systems, running two copies of the benchmark program simultaneously resulted in each of the benchmark programs taking twice as long to complete. The 4-processor 670MP system, however, took only 50% longer to complete two simultaneous jobs. This result suggested to us that a multiprocessor system would give us good performance and acceptable response times even under heavy loads. Our experience over the last six years has proven this to be true.

Reducing Disk Space Requirements

In addition to being seen as a "performance pig," SAS software is also sometimes seen (wrongly) as a "disk hog." SAS software is not the problem - your data sets are. One way to reduce the disk storage requirements of your data sets is to compress them using the COMPRESS data set option.

In our experience, compression typically reduces the size of our data sets by 50%. Since our system has more than one processor, the added processing required to compress and decompress data sets as they are written to and read from disk has a negligible impact on performance. Compressed data sets also occupy less physical space on the disk, requiring less system time to perform disk reads and writes.

Compression, however, does not come without a cost. Observations in a compressed data set cannot be accessed by observation number. Also, the COMPRESS option can, in some cases, make your data sets larger. This will occur if your data sets contain only unique values, such as a list of social security numbers.

Setting SAS System Options

There are several SAS system options which can have an impact on system performance. Among these are the BUFNO and BUFSIZE options, the MEMSIZE and SORTSIZE options, and the previously mentioned COMPRESS option.

The BUFNO and BUFSIZE options specify the number and size of the data buffers that the SAS System will use when creating data sets. Our experimentation with these values has shown that the default values of 1 for BUFNO (meaning that SAS will allocate only one data buffer) and 0 for BUFSIZE work the best for our system. A BUFSIZE of 0 tells SAS to select the optimal buffer size.

The MEMSIZE and SORTSIZE options specify the total amount of memory that can be used by the SAS System and the maximum amount of memory that can be used for sorting. On our system, with 192MB of RAM installed, we use values of 128MB for each of these options.

Setting the COMPRESS option to YES in

the system-wide *config.sas* file ensures that all SAS data sets will be compressed by default, which can save a considerable amount of disk space. This option can be overridden in individual SAS programs, if necessary.

Setting Process Priorities

The Unix operating system offers the *nice* utility program which is used to raise or lower the priority of a given process. We encourage users who submit SAS batch jobs (which run in the background) to start the processes at a lower priority by using *nice*. Jobs can also be *niced* by the superuser after they start using the *renice* command.

Processes that are executed using *nice* will yield the processor to other processes that are ready to run. This will result in decreased performance for jobs submitted using *nice*, and better performance for all other jobs running at the system default priority level. The system administrator can also raise the priority of a process using *nice*.

Eliminating Unnecessary Processes

One can often improve the performance of a Unix system by eliminating some of the processes that start automatically at boot time. For example, the SunOS system accounting program *sa* is normally started when the system boots. If you do not bill for CPU time on your Unix system, this process is unnecessary and can be eliminated. Depending on how your system is configured, it may be possible to eliminate other processes such as *routed*, *quotad*, and *sendmail*.

Monitoring Performance

The performance of our system has been monitored since its installation six years ago. Every day at 5:20 P.M. we take a

snapshot of overall system performance (using the SunOS *vmstat* utility) and log the resulting data to a text file. The data can then be graphed using SAS/GRAPH® software (see Figure 1).

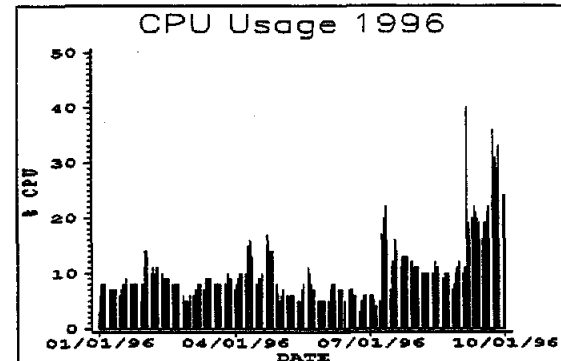


Figure 1

Having a historical record of system performance makes it easier to determine the cause of performance problems when they arise. Performance problems can arise long after initial system configuration due to the installation of operating system upgrades, patches, or new applications.

After our 670MP system had been running for four years, a new multiuser Ingres application was deployed. The new application had a noticeable impact on system performance. Initially the blame for the decrease in performance was placed on SAS, however, when we looked at the chart of performance for that year, there was an obvious increase in CPU cycles at two points - first, when the Ingres application was first brought online, and second, when the *tmpfs* filesystem (initially configured to improve SAS performance) was eliminated to make more memory available for Ingres.

Having a historical view of system performance made it possible to pinpoint when the performance decrease began, and when this was compared to system change logs, it was possible to pinpoint the specific configuration changes that were responsible for it.

Conclusion

There are many steps that a system administrator can take to maximize SAS software performance under the Unix operating system. These steps include the selection of an appropriate system with the help of a reliable benchmark and proper configuration of the hardware, the operating system, and the SAS System. Note that in many cases, maximizing the performance of your system will not require the purchase of additional hardware, and significant improvements can be made by utilizing built-in features of both the SAS System and the Unix operating system.

References

SAS Institute Inc. (1990), *SAS Companion for the Unix Environment and Derivatives, Version 6, First Edition*, Cary, NC: SAS Institute Inc.

Mike Loukides (1991), *System Performance Tuning*, Sebastopol, CA: O'Reilly and Associates, Inc.

SMCC Technical Marketing (1993), *Sun Performance Tuning*, Mountain View, CA: Sun Microsystems, Inc.

Sun Microsystems Inc. (1990), *SunOS Reference Manual*, Mountain View, CA: Sun Microsystems, Inc.

Acknowledgments

SAS and SAS/GRAPH software are registered trademarks or trademarks of the SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

IBM and TSO are registered trademarks or trademarks of International Business Machines Corporation. ® indicates USA registration.

Other brand and product names are

registered trademarks or trademarks of their respective companies.

Authors

Daniel McLaren
Henry Ford Health System
Department of Biostatistics and Research
Epidemiology
1 Ford Place, Suite 3C
Detroit, MI 48202
(313)874-6706
Email: dmclare1@hfhs.org

George W. Divine, PhD.
Henry Ford Health System
Department of Biostatistics and Research
Epidemiology
1 Ford Place, Suite 3E
Detroit, MI 48202
(313)874-6724
Email: gdivine1@hfhs.org