

June 2005



**Hints and Tips for Running SAS<sup>®</sup>  
Software on an IBM<sup>®</sup>@server  
<sup>®</sup>pSeries<sup>®</sup> server running AIX 5L<sup>®</sup>**

IBM<sup>®</sup> Systems and Technology Group

Abstract
Introduction
General Suggestions
Detailed Suggestions
Setting up the AIX Kernel
Choosing the File System
Storage Considerations
I/O Considerations
Direct I/O
Performance Tuning
Conclusion
App 1: Reading the SAS Log
App 2: AIX Performance Tools
App 3: Metadata Hints and Tips

## **Abstract**

This paper presents practical information for running SAS 9 on AIX on an IBM@server pSeries server. Some information is also relevant to SAS 8.2. A high level list is first presented with subsequent sections providing supporting details and references. The objective of this document is to provide system administrators with practical hints and tips to improve the performance of SAS 9 on AIX. It is not intended to provide in-depth or advanced tuning information but merely to introduce some of the concepts and indicate where further documentation can be found.<sup>1</sup>

## **Introduction**

SAS is being installed on more and more IBM@server pSeries servers because of the platform's industry leading price/performance and scalability. Default kernel settings for AIX 5.1 and 5.2 are for general workloads and are not necessarily optimal for SAS. For instance, long sequential reads and writes are generally the norm for SAS programs, an access pattern that does not perform well with default settings for AIX 5.1 and 5.2. For AIX 5.3 the default kernel settings were adjusted upward towards enterprise level workloads and require fewer changes to perform well with SAS.

At this writing the best environment for running SAS is AIX 5.3 on POWER5 processors, but the success of any implementation of any software is based on carefully planning, and setting up the environment to suit the needs of the users and the workload the users present. The following general suggestions and subsequent explanations provide a good starting point for creating just such an environment for SAS 9.

## **General Suggestions**

The following items comprise a list of general suggestions for installing and running SAS 9 on pSeries servers running AIX. Further details for each suggestion can be found in subsequent sections of this paper.

- Use the latest version of AIX and keep the OS up to date with the latest maintenance level and fixes from IBM. At this time the latest version of AIX is AIX 5.3. Significant I/O performance improvements have been made in AIX 5.3 over AIX 5.1 and 5.2. Many customers report a 10% or greater improvement in I/O throughput just by upgrading to AIX 5.3. Also, if you manage the server with a Hardware Management Console (HMC), keep the HMC up to date. You can use the Microcode Survey and Update Tools at <http://techsupport.services.ibm.com/server/mdownload/mcodetools.html> to keep the server and device firmware up to date. Use the pSeries Subscription Service <https://techsupport.services.ibm.com/server/pseries.subscriptionSvc> to be notified of the release of AIX, HMC and microcode updates and fixes.

---

<sup>1</sup> This paper is not intended to provide information on writing SAS® applications. Although it is true that good code has a profound impact on performance, there are papers and examples available on this subject at: <http://www.sas.com>.

Furthermore, this paper is not meant to be a presales paper. Presales help is available by contacting the IBM International Competency Center at SAS Institute: [ibmsas@us.ibm.com](mailto:ibmsas@us.ibm.com). A copy of the current IBM/SAS sizing questionnaire and instructions can be found at: <http://www-1.ibm.com/support/techdocs/atsmastr.nsf/PubAllNum/PRS512>

- During AIX installation on 64-bit platforms select the 64-bit kernel to be installed. The 64-bit kernel automatically and simultaneously runs 64-bit programs like SAS 9 in 64-bit mode and 32-bit programs like SAS 8.2 in 32-bit mode. The 64-bit kernel takes advantage of the 64-bit hardware and is compiled at a higher level of optimization and is more scaleable than the 32-bit kernel. (For more information please see the section called "Setting Up the AIX Kernel")
- Use JFS2® file systems. In almost all cases JFS2 will out perform JFS®. (For more information please see the section called "Choosing the File Systems")
- Set up the I/O subsystem with as many physical disks as possible (do not set them up as a single volume/mount point). Use as many spindles and controllers as possible (many small disks are better than a few large disks).
- Keep SAS WORK and UTILLOC areas and data storage areas on separate storage hardware using separate disks, controllers and I/O channels if possible. Implement striping, and consider your needs for high availability. The more I/O channels that are provided, the better. (For more information please see the section called "Storage Considerations")
- In general sequential processing of data sets is the norm for SAS programs. If this is the case for your use of SAS then tuning AIX for more efficient sequential I/O will shorten the runtime of SAS programs. (For more details see the section called "I/O Considerations")
- Individual SAS programs will often read in a raw data file or a data set only once and not access that data again. The AIX Virtual Memory Manager (VMM) does not have information about the reuse potential of files so the VMM's default action is to use available main memory as a file system cache to keep a copy of the file data available for fast access. Caching file data uses memory page frames and if the number of free page frames falls below a certain number then the Least Recently Used (LRU) daemons will run to free up page frames. The LRU daemons consume CPU cycles and memory and I/O bandwidth when they run. As an optimization AIX has release-behind-read and release-behind-write features that can be configured on a per file system basis. These options allow the VMM to release file pages from the file system buffer cache right after SAS has read or written the file pages. (For more details see the section called "I/O Considerations").
- Syncing up the SAS option BUFSIZE with disk subsystem stripe size, LVM stripe size and VMM read-ahead increments can result in fewer physical I/Os per read and write request which improves performance. In other words set up stripe size, BUFSIZE, and read ahead parameters to have a common whole-number factor (For more details see the section called "I/O Considerations").
- Consider using direct I/O (DIO) if CPU and memory resources are so constrained that freeing up CPU and memory resources warrants the tradeoffs. The tradeoffs are that files accessed with DIO are not cached so physical I/O must take place for every access and VMM read ahead and write behind are not available. If you determine that DIO is warranted then put the SAS datasets that you want to access with DIO into separate directories and remount those directories for DIO or use a namefs mount with DIO. SAS datasets produced with earlier versions of SAS must also be properly conditioned for direct I/O to work efficiently. (For more details see the section called "Direct I/O")
- Since each SAS job runs in its own SAS process one can manage the workload to ensure that more critical SAS processes are guaranteed certain resources. Ways to manage the workload to ensure CPU and memory resources are available to those programs that need them are: Using the NICE and RENICE commands or IBM

Workload Manager to prioritize which processes get CPU time and how much time they will receive, or using Logical Partitioning (LPAR) strategies to separate conflicting workloads and balance the system.

## ***Detailed Suggestions***

### **Choosing the File System**

The Journalled File System (JFS) was introduced with the initial release of AIX Version 3.1. If the server will primarily handle thousands of files smaller than 128 KB in size (a common situation when serving web pages or sub-setting databases) then JFS file systems created with the appropriate parameters might be more efficient.

For SAS WORK areas and other temporary data areas, where transient data and availability are not an issue, the JFS mount option "no integrity" can be used for file systems that can easily be reloaded or recreated. This will prevent system time and disk-I/O being spent on synchronous writes to the "jfslog". Note this is only available for JFS and if there is a power failure or the system crashes then the integrity of the file system's meta-data cannot be guaranteed (as it is when a "jfslog" is used). Also, JFS does not scale as well as JFS2.

With AIX 5L a second file system for AIX, called Enhanced Journalled File System (JFS2), was introduced. JFS2 allows larger files to be processed and has many favorable performance qualities that make it the proper choice for new installations of SAS where files greater than 128 KB will be used. Use the 64-bit kernel for AIX with JFS2 (JFS2 is supported on the 32-bit kernel under AIX but results in lower performance<sup>2</sup>).

Another file system that might be used is the Veritas® File System (VxFS) that can run as a standalone product or with Veritas Volume Manager (VxVM) or other volume managers. VxFS can optimize I/O and is designed to work with standard volume managers. JFS and JFS2 are designed to run on the AIX Logical Volume Manager (LVM) that is built into the base of the AIX operating system.

The Veritas file system has additional features not found in JFS and JFS2, however JFS and JFS2 by virtue of being leaner, are typically faster and provide a robust environment for the SAS user.

Avoid NFS access of large data sets if possible. NFS mount points are transparent, so users might not realize that they are accessing large data files across a network. Transferring large volumes of data over networks might limit application performance and cause network congestion. On a similar note, avoid the need for many SAS/CONNECT™ sessions that require large data transfers because they can cause I/O and network bottlenecks.

Additional information on file systems can be found at:

Quick Reference: AIX Journalled File Systems and Veritas File System

---

<sup>2</sup> Even though JFS2 provides significant scalability improvements over JFS, its full potential will only be realized with the 64-bit kernel. The 64-bit kernel provides a better environment for running JFS2 because 32-bit addressing limits scaling potential. The combination of the 64-bit kernel and JFS2 is designed to perform much better than any other combination of file system and kernel with respect to file system performance. If the 32-bit kernel must be used then JFS might be a better choice.

<http://www.redbooks.ibm.com/redpapers/pdfs/redp0102.pdf>

Problem Solving and Troubleshooting in AIX 5L

<http://www.redbooks.ibm.com/pubs/pdfs/redbooks/sg245496.pdf>

## Storage Considerations

Place operating system, SAS, paging space and user home directories on separate volumes. Multiple users sharing the same SAS WORK area can create disk contention. Either increase the number of I/O paths (stripe to different storage cabinets if necessary), or set up separate SAS WORK areas each with their own disks (with separate I/O channels if possible).

For high availability, the system and SAS volumes can be mirrored and striped (RAID1+0). Other permanent data can be stored using RAID5. SAS WORK, and other temporary files can be configured to use RAID0 (disk striping) unless there is a need for high availability (if so use RAID1+0). It is preferable to stripe data (both permanent and temporary) across multiple disks. If it is not possible to stripe, then place input, output and SAS WORK on separate volumes. If you can stripe only some of your storage, stripe your SAS WORK storage. Synchronize BUFSIZE with stripe-size. If you are using disk striping for the first time try 64KB or 128 KB as the stripe size and set BUFSIZE to the product of stripe-size and the number of data disks you are striping across. Use a power of 2 for the number of data disks in the array.

**Serial Storage Architecture Devices** - If your SAS storage is located on an IBM Serial Storage Architecture (SSA) device such as the IBM 7133 disk subsystem, then:

1. Allocate input, output, and SAS WORK directories on separate SSA logical disks with each SSA logical disk made from a striped array of SSA physical disks.
2. Use multiple file systems. This distributes file system log activity across more of the storage server, minimizing hot spots.

**Enterprise Storage Server** - If your SAS storage is located on an IBM Enterprise Storage Server (ESS), then:

1. Distribute your SAS storage evenly across the entire ESS.
2. Do an additional level of striping across the arrays and controllers internal to the ESS. Because Enterprise Storage Servers employ hardware RAID 5 internally, data is automatically striped; however, performance might be improved even more by doing an additional level of striping across the arrays and controllers internal to the ESS.
3. Use multipathing: a minimum of two Ultra SCSI or two Fibre Channel connections to each ESS volume is recommended.

Though performance is important, one should also consider high availability. All in all RAID 5 is recommended as a good compromise between high performance, high availability, and cost.

For more information about storage, see the following documents:

Configuration Options using IBM SSA Storage for SAS:

<http://www.sas.com/partners/directory/ibm/storagessa.pdf>

Maximizing Performance on IBM Enterprise Storage Servers:

<http://www.sas.com/partners/directory/ibm/storageess.pdf>

High Availability Storage on IBM Enterprise Storage Servers:

<http://www.sas.com/partners/directory/ibm/storageavail.pdf>

Managing SAS Storage on IBM Enterprise Storage Servers:

<http://www.sas.com/partners/directory/ibm/sasstorage.pdf>

## I/O Considerations

**Read Ahead** - For SAS jobs that do large sequential reads, one can use vmtune to adjust read ahead (Beginning in AIX 5.2, this command is a compatibility script that calls vmo and ioo). For AIX 5.3 use the vmo and ioo commands directly or through the smit (system management interface tool) command. The purpose of the VMM read ahead algorithm is to improve the performance of sequential file access. If the VMM detects that a process has read two pages of a Journalled File System (JFS) or Enhanced JFS (JFS2) file sequentially, then the VMM will also schedule a number of subsequent file pages to be read into file cache memory. Note that a page is 4K bytes.

For JFS the initial number of pages is specified with the minpgahead parameter and the maximum number of pages is set with the maxpgahead parameter. The default value is two for minpgahead and eight for maxpgahead. The analogous parameters for JFS2 are minPageReadAhead and maxPageReadAhead. System administrators can modify these parameters with the VMTUNE parameter or for AIX 5.2 and above with:

**ioo -o j2\_minPageReadAhead**=Number to set the minimum number of pages to read ahead for JFS2

**ioo -o j2\_maxPageReadAhead**=Number to set the maximum number of pages to read ahead for JFS2

**ioo -o minpgahead**=Number of pages with which sequential read-ahead starts

**ioo -o maxpgahead**=Number to set the minimum number of pages to be read ahead

See the AIX 5L Version 5.3 Performance Management Guide

<http://publib.boulder.ibm.com/infocenter/pseries/topic/com.ibm.aix.doc/aixbman/prftungd/prftungd.htm> for more details. For AIX 5.3 the default parameters are set

higher than in 5.1 and 5.2. See AIX 5L Practical Performance Tools and Tuning Guide <http://www.redbooks.ibm.com/redbooks/pdfs/sg246478.pdf>.

As the sequential read pattern continues, the VMM continues to schedule subsequent pages of file data to be read as long as they are not already in the file cache. The number of pages is doubled for each successive read until it reaches the maximum number specified (Set with maxpgahead for JFS and maxPageReadAhead for JFS2). As with any kind of tuning, knowledge of the application is important, and a lot of small tweaks and much observation are the order of the day. Adjustments to maxpgahead might even reduce throughput if it turns out that the paging is occurring during random reads. The vmtune command can only be executed by root. Changes made by the vmtune command last until the next reboot of the system. If a permanent change in VMM parameters is needed, an appropriate vmtune command can be included in **/etc/inittab** or vmo and ioo can be used.

**Read Sequential, Write Sequential, and Release Behind** - SAS applications and solutions are unusual in that there is a higher likelihood that there will be a great deal of sequential reading and writing, and that often, once read, the data will not be read again. If this is the access pattern then mounting the file systems for release behind and read/write sequential or some variation thereof might increase performance. The various mount options are: release behind read/write sequential (-rbrw), release

behind and read sequential (-rbr), or release behind and write sequential (-rbw). Use the mount option "rbr" (release behind read) for file systems with large files (they are large if they would occupy a significant portion of the file cache) that are read once by only one job and do not need to be cached in the file cache. Once the requesting program is given the file data the VMM storage for the file is released from the file cache. Without this option the file data would continue to occupy memory with no benefit of reuse. Use the mount option "rbw" (release behind write) for file systems with large files that are written sequentially and require very little, if any, additional access during the rest of the job. Once the VMM passes the file data to the disk-I/O-subsystem the VMM storage for the file is released from the file cache. Without this option the file data would continue to occupy memory with no benefit of reuse. As always, one size does not fit all, so monitor performance after this change to see if performance is improved or diminished. More information can be found at:

AIX 5L Performance Tools Handbook

<http://www.redbooks.ibm.com/pubs/pdfs/redbooks/sg246039.pdf>

**Syncing up Transfer Size** - If one is processing large files (files larger than the maximum transfer size) all the data should be transferred in compatibly sized "chunks". Make sure that BUFSIZE is equal to or evenly divisible by the transfer size (single disk) or stripe size (disk array) of the disk subsystem.

## Direct I/O

Direct I/O is a file access method that was first introduced in AIX 4.3 and has been available for all later releases of AIX. When an application accesses files that are opened for direct I/O, AIX uses the pSeries DMA hardware to transfer file data directly between disk adapters and the SAS process I/O buffer without using an intermediate file cache.

As long as CPU and memory resources are available, the best I/O strategy for most workloads is the default method of file access (here referred to as file caching). This is because with file caching, AIX enables CPU and memory resources to be used to minimize program run time. If, however, CPU or memory resources are severely over committed, some files including SAS datasets can be opened for direct I/O to free up the CPU and memory resources used for file caching to improve CPU and memory availability and reduce contention for the file cache.

Current versions of SAS products do not provide an option for SAS programs to explicitly set the O\_DIRECT flag in the AIX open() system call, thus they cannot specify that individual files be opened for direct I/O. AIX does however provide a mechanism to force files to be opened for direct I/O. With the JFS and JFS2 file systems, directories can be mounted with the direct I/O mount option -o dio specified in the mount command. Any files opened in these directories are then opened for direct I/O.

To use direct I/O for SAS with older versions of AIX version 5.1 should at least be at Maintenance Level 4 and AIX 5.2 should at least be at Maintenance Level 2 (SAS was unable to use direct I/O with earlier versions of AIX because of buffer alignment characteristics that caused direct I/O to be canceled by the filesystem code).

Though SAS input can come from many sources only SAS datasets or SAS 8.2 utility datasets should be considered for direct I/O - if the input were a raw data file or a database, SAS wouldn't perform the large sequential reads that profit from avoiding the file cache. Also, direct I/O would be canceled for the reads and writes and a degraded form of file caching would be used.

Filesystem Format	File Offset Alignment	Maximum Transfer Size	Minimum Transfer Size
JFS fixed, 4 K block	4KB	2MB	4KB
JFS fragmented	4KB	2MB	4KB
JFS compressed*	N/A	N/A	N/A
JFS large file enabled**	128KB	2MB	128KB
JFS2	4KB	4GB	4KB

**Table 1 – DIO offsets and alignments for JFS and JFS2 Filesystem Formats**

\*JFS Compressed will not work with direct I/O

\*\* JFS with Large File Enabled will only work with SAS 9.1 and above

For direct I/O to work, files must be accessed with the correct offset alignment and transfer size according to the filesystem used (see Table 1).

The file-offset alignment used by read and write system calls must be evenly divisible by the value in the table (4KB for JFS fixed, JFS fragmented, and JFS2 and 128 KB for JFS large file enabled). The transfer size for a read or write system call must be greater than, and evenly divisible by the minimum transfer size and no greater than the maximum transfer size. Failure to meet these requirements will cause direct I/O to be canceled for the transfer and cause file reads and writes to use a degraded form of file I/O that is slow and inefficient.

For SAS datasets, the file-offset alignment is determined by the size of the SAS dataset header. The header size in a SAS dataset is 8K, which satisfies the offset alignment requirement for direct I/O to function with JFS and JFS2 (the exception is JFS with large-file support enabled, which will be addressed later).

In SAS, transfer size is determined by a SAS parameter called "BUFSIZE". BUFSIZE is a parameter that can be set globally in the SAS config file, and overwritten or set within individual SAS programs. The SAS dataset will be structured for the specific BUFSIZE at the time that it is created. To meet the requirements for direct I/O BUFSIZE should be evenly divisible by the minimum write length, and no larger than the maximum write length. Other considerations should also be taken into account when setting BUFSIZE such as the fact that direct I/O cannot take advantage of the VMM read ahead algorithm. Read ahead can be extremely effective at reducing runtime for programs that access files sequentially. Read ahead reduces runtime by issuing larger and larger read-requests and by overlapping reads of future blocks with application processing.

If BUFSIZE is set to at least 128KB for datasets mounted for direct I/O it will cause larger transfers to take place, compensating in some part for being unable to use read ahead.

If bufsize is set to less than 128 KB then performance will most likely be dismal. SAS datasets created with BUFSIZE set to 128K will work for all of the file formats.

The BUFSIZE of a dataset can be determined from a SAS program with a procedure called PROC CONTENTS.

Example:

```
proc contents data=libname.dataset;
run;
```

If SAS datasets were created with BUFSIZE set to anything less than 128K, or not evenly divisible by the minimum transfer size (see Table 1) then a new SAS dataset must be created and the contents of the old dataset must be copied in. The following SAS data step explicitly specifies the BUFSIZE and creates a new dataset. It then copies the data from the old dataset that was created with the incorrect bufsize size into the new dataset:

```
data new (BUFSIZE=128k)
  set libname.old;
run;
```

The output of this SAS data step will be a new SAS dataset containing the same data, but structured for direct I/O.

For direct I/O to work under JFS with large file enabled a specific modification was made to SAS 9.1 so that it would meet the “Alignment Offset” requirement of 128K. Prior releases of SAS will not meet the alignment offset for JFS large file enabled and thus cannot access this filesystem format with direct I/O. A third parameter, “offset”, was added to the SAS 9.1 System libname directive (A directive within a SAS program that specifies where data areas are located on the system). The “offset” parameter indicates how far after the header the actual data should be (in essence it “pads” the header to the value specified. The default header is 8K). The “offset” parameter should be used with the undocumented “minpagesize” and “optincrement” parameters as follows:

```
libname foo '.' minpagesize=128k optincrement=128k offset=128k;
```

Because SAS programs cannot explicitly set the O\_DIRECT flag in the AIX open() system call, the file system must be mounted with -o dio: the direct I/O mount option.

Once the datasets have been conditioned, and the directories that contain the datasets have been mounted with the direct I/O mount option, I/O requests by SAS programs for data within these datasets will occur without using the file cache.

## ***Performance Monitoring and Tuning***

Once SAS has been installed the server should be monitored and tuned for performance. A well-tuned system balances CPU, I/O, and memory resources to meet the needs of application requests. Focusing on performance in one area without regard to the other areas can actually diminish overall performance.

If one approaches performance tuning in a systematic way, correctly sizing the system initially, and taking care to monitor and optimize processor utilization, memory usage, and input/output, SAS will run quite nicely on the pSeries Server. It is important to remember that performance tuning is a process, each subsystem (CPU, Memory, and I/O) should be tuned progressively, and tuning in one area will impact the performance of another, thus several iterations of the process might be necessary before one achieves the desired result. Though it is not the purpose of this

paper to provide in depth tuning information, a brief explanation of the SAS log and a survey of the tools available for monitoring AIX are included in appendix 1 and 2.

### ***Conclusion***

When setting up an eServer pSeries the recommendations covered in this paper are a good starting point for a successful SAS implementation. These recommendations include using the AIX 64-bit kernel, using JFS2 (or, if files will be greater than 128 KB, JFS), setting up the I/O subsystem with a large number of disks and as many spindles and controllers as possible, keeping SAS WORK areas and data storage areas segregated from each other and using separate I/O channels if possible, implementing striping, setting up VMM parameters for long sequential reads and writes, syncing up transfer size, and managing the workload. These are the measures for creating a winning environment for SAS applications.

## Appendix 1: Reading the SAS Log

It is often the users who first discovers performance bottlenecks and brings them to the attention of the system administrator.

When it comes to SAS, the user might come waving a set of log files showing that, indeed, something has caused the execution time to increase. The administrator can use these logs to do a preliminary diagnosis. To that end, here is a brief explanation of the SAS log and what can be determined from it.

If the FULLSTIMER option is turned on, the simple stats that appear in the SAS log will be greatly expanded. They will show up as tables similar to the one below (output is specific to AIX) and appear after each step as well as at the end of the SAS log.

**NOTE: The SAS System used:**

<b>real time</b>	<b>1:16.76 seconds</b>
<b>user cpu time</b>	<b>1:12.35 seconds</b>
<b>system cpu time</b>	<b>2.45 seconds</b>
<b>Memory</b>	<b>1887k</b>
<b>Page Faults</b>	<b>0</b>
<b>Page Reclaims</b>	<b>1537</b>
<b>Page Swaps</b>	<b>0</b>
<b>Voluntary Context Switches</b>	<b>823</b>
<b>Involuntary Context Switches</b>	<b>7512</b>
<b>Block Input Operations</b>	<b>10</b>
<b>Block Output Operations</b>	<b>12</b>

If this detailed information is not in the SAS Log then the FULLSTIMER has not been turned on. The FULLSTIMER can be turned on within a SAS program with the following statement:

```
OPTIONS FULLSTIMER;
```

It can also be turned on from the command line:

```
$(SAS_INSTALL_DIR)/sas myprog.sas -fullstimer
```

There will be a similar table for every step within the SAS job, so the SAS log can help pinpoint performance problem down to the step.

**An explanation of each of the items in the SAS log** -These entries have the following definitions:

- **real time:** Elapsed wall-clock time spent waiting for a job or step.
- **user cpu time:** The total amount of CPU time spent running in user mode.
- **system cpu time:** The total amount of CPU time spent in system mode running on behalf of the process.
- **Memory:** The maximum amount, in kilobytes, of memory used for the job or step (not including the memory the SAS session is consuming).
- **Page Faults:** The number of page faults that resulted in real I/O requests. Also known as major page faults (I/O activity occurred).
- **Page Reclaims:** The number of page faults serviced without any I/O activity. I/O activity is avoided by reclaiming a page frame from the list of pages awaiting reallocation. Also known as minor page faults (no I/O activity occurred).

- **Page Swaps:** The number of times a process was swapped out of main memory (Always zero on AIX).
- **Voluntary Context Switches:** The number of times the job or step released its CPU time-slice because it made a system call that could not return immediately. These system calls are I/O requests or wait-for-a-resource type of requests.
- **Involuntary Context Switches:** The number of times the job or step released its CPU time-slice because it completely used a 10 ms time slice or was preempted by a higher priority thread.
- **Block Input Operations:** The number of “chunks” of data read. These are I/O operations to read the data into memory for usage. Not all reads have to utilize an I/O operation since the page being requested might still be cached in memory from previous reads. The size of each chunk of data is based on the SAS “bufsize” parameter.
- **Block Output Operations:** The number of “chunks” of data written. These are the same as block input operations except that they pertain to the writes to files. As in the case of block input operations, not all block outputs will cause an I/O operation. Some files might still be cached in memory. Again the size of each “chunk” of data is based on “bufsize”.

**Reading the log** - The first thing to look for in the SAS log is the existence of errors that prevented the completion of the program, or any errors reading files. If any errors are present, they should be resolved before attempting to glean any information about performance.

If there are no errors reported in the SAS Log, compare real time and CPU time (the sum of the user CPU time and System CPU time). If the CPU time is significantly less than the real time (Say 20% of real time, just to put a number on it), then either the process is being blocked or, if the Memory listing appears high, adjustments might need to be made to virtual memory settings. If the process is being blocked, it is either because it has been preempted by a higher-priority task or it is waiting for a system call (probably I/O) to complete.

Next, look at the involuntary context switches. If these are consistently high (across multiple jobs and steps) it would be another indication that the CPU is under heavy load.

If the system is not CPU bound, then the next thing to look at in the SAS log is the number of Voluntary Context Switches. These indicate the number of times the procedure voluntarily gives up its time-slice waiting on an external event such as I/O. A high number of voluntary context switches compared to involuntary context switches suggest inefficient I/O possibly due to paging to page space. More information on solving SAS performance problems can be found at:

<http://support.sas.com/rnd/papers/sugi27/SolvingPerformance.pdf>.

## Appendix 2: AIX Performance Tools

AIX has several tools for monitoring CPU, memory and I/O performance. Among the best of these are: vmstat, iostat, sar, time, ps, tprof and topas. The best freeware tool for AIX performance monitoring is nmon.

### A Procedure for collecting an nmon trace reflecting a representative time slice of application workload

1. Point your browser to the following link to download NMON at the most current release:

[http://www-106.ibm.com/developerworks/eserver/articles/analyze\\_aix/index.html](http://www-106.ibm.com/developerworks/eserver/articles/analyze_aix/index.html)

2. Acknowledge the freeware disclaimers:

```
-----> International License Agreement for Non-Warranted Programs
```

3. Download to PC client, and ftp this file in binary mode to target systems;

```
nmon ftpserv ----> PC ----> AIX
```

or, direct access from AIX browser ;

```
nmon ftpserv ----> AIX
```

4. Uncompress (uncompress -v <filename.tar.Z) and extract using tar (**tar -xvf <filename.tar>**). nmon file contents into any directory, I suggest an nmon directory in sysadm \$HOME workspace. Estimated file size for download time: **nmon9a.tar.Z is ~700KB. Extracted is ~5MB**

5. To collect data, execute either nmon, or nmon64, depending on the version of the OS kernel running on target system a/o partition.

- a. To check the OS version and service levels, use following commands:

```
oslevel -r returns 5300-02 AIX Release : 5300 Maintenance Level:02
```

- b. To check the OS kernel level (requires root permission):

```
bootinfo -K returns 64 (64-bit kernel) or 32 (32-bit kernel)
```

6. The following command sequence will create a file in whatever directory nmon is executed from (ie; \$HOME/nmon will create an output file, default fn is <hostid\_date\_time>.nmon, at 30 sec intervals, for 300 intervals, or precisely 2.5 hrs, including the top resource consuming processes during measured intervals. The -s, and -c flags should be adjusted to accommodate for workload timing profile for trace activity. If we are only monitoring a 15m time slice, maybe 5 or 10 sec intervals are more appropriate. Have to flex based on the circumstances. Produce and deliver results to IBM for trace analysis)

- a. 32 bit AIX kernel : **nmon -f -s 30 -c 300 -t**
- b. 64 bit AIX kernel : **nmon64 -f -s 30 -c 300 -t**

NOTE : To use NMON in interactive mode, type **nmon**, or **nmon64**, using a uid with system level reporting authority, and an interactive console is presented, with on-screen help mode. Reporting is by toggle mode, hitting **c** reports CPU utilization, hitting it again turns it off. Hit **m** to report memory, **d** for disk, etc. Detailed help is also available in `read.me` in distribution pkg, and from the **nmon** interface by hitting the **h** key.

### ***Appendix 3: Hints and Tips for Setting Up the Metadata Server on an IBM @server pSeries***

The SAS Metadata Server is a centralized repository for storing and managing enterprise metadata. Metadata is data about data. It is information about the data sources, how data were derived, business rules, and access authorizations).

There are two ways to set up the metadata server for the single pSeries server installation of SAS. The first is to utilize logical partitioning (LPAR), and the other is to just install the metadata server and SAS in one partition. The later method might create some performance problems, but these can be alleviated by binding two of the processors to the metadata server. Binding does not prevent other processes from being dispatched on the processor on which you bound your process. Binding is different from partitioning. Without Workload Manager (WLM), introduced in AIX 4.3.3, it is not possible to dedicate a set of processors to a specific workload and another set of processors to another workload. Therefore, a higher priority process might be dispatched on the processor where you bound your process.

What this means for the metadata server is that it can be limited to two processors but SAS processes can still use those processors.

Use the `bindprocessor` command to bind or unbind the kernel threads of a process to a processor. Root authority is necessary to bind or unbind threads in processes that you do not own.

Note: The `bindprocessor` command is meant for multiprocessor systems. Although it will also work on uniprocessor systems, binding has no effect on such systems.

To query the available processors, run the following:

```
# bindprocessor -q
```

The available processors are: 0 1 2 3

The output shows the logical processor numbers for the available processors, which are used with the `bindprocessor` command as will be seen.

To bind a process whose PID is 14596 to processor 1, run the following:

```
# bindprocessor 14596 1
```

No return message is given if the command was successful. To verify if a process is bound or unbound to a processor, use the `ps -mo THREAD` command as explained in Using the `ps` Command:

```
# ps -mo THREAD
```

© Copyright IBM Corporation 2005  
IBM Corporation  
Marketing Communications  
Systems Group  
Route 100  
Somers, New York 10589

Produced in the United States

June 2005

All Rights Reserved

No part of this document may be reproduced or transmitted in any form without written permission from IBM Corporation.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This information could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or programs(s) at any time without notice.

The performance data contained herein was obtained in a controlled, isolated environment. Actual results obtained in other operating environments might vary significantly. While IBM has reviewed each item for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead. It is the user's responsibility to evaluate and verify the operation of any non-IBM product, program or service.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice and represent goals and objectives only.

The information provided in this document is distributed "AS IS" without any warranty, either express or implied. IBM EXPRESSLY DISCLAIMS any warranties of merchantability, fitness for a particular purpose OR non-INFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. IBM is not responsible for the performance or interoperability of any non-IBM products discussed herein.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

#### Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both: AIX, AIX 5L, DB2, IBM, Enterprise Storage Server, eServer, FAST, i5/OS, iSeries, i5, Micro-Partitioning, OpenPower, OS/390, OS/400, POWER5, pSeries, p5, RS/, S/390, Tivoli, TotalStorage, xSeries, z/OS, xSeries, zSeries.

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries.

Other company, product or service names might be trademarks or service marks of others.

