Setting up and tuning direct IO for SAS 9 on AIX

Frank Bartucca
IBM eServer Solutions Enablement
November 2005

Table of contents

Abstract

The User Interface Design and Performance Analysis department, in the Analytical Solutions Division at SAS®, requested an IBM® AIX® system for the purpose of setting up a large data mart to test the performance of extract, transform, and load (ETL) flows. The SAS R&D Data Center provided an IBM pSeries® p650 system running the IBM AIX 5L™ Version 5.3 operating system with a host name of "lemans." The pSeries technical consultant onsite at SAS assisted with the setup and tuning of lemans. The results obtained are documented in this paper.

Introduction

Whether economic conditions are forcing a rediscovery of economy of scale, or pressure is being placed on IT shops to provide systems that support increasing levels of data processing under a tight budget, a trend has emerged towards consolidating servers and workloads onto larger servers. As a result, SAS programs in large multiuser ETL environments are now processing high volumes of data. It has been noted that performance in these conditions greatly depends on the tuning and optimization of server I/O subsystems and application I/O options. This paper presents practical information that SAS users and system administrators can use to tune their own pSeries systems.

Steps of the tuning process

The tuning effort described in this paper was structured as follows:

- The major features of the server were understood.
- Then the initial configuration and setup were determined.
- Next, the upper limit of the I/O subsystem's performance was determined, and a SAS workload representative of SAS jobs was selected and run on lemans.
- For the first run, I/O throughput was measured using default settings to establish a baseline.
- After that, the rest of the tests involved iterative tuning of various SAS and AIX options to yield a job that would be less disruptive to the overall performance of a multiuser system.

Concepts

The following concepts will be discussed in this paper. It is advised that the reader have a base understanding of each before continuing.

- Cached and direct I/O
- I/O buffering
- SAS I/O-related options
- AIX tuning parameters
- File system buffer cache
- Logical volumes
- Performance monitoring
- Storage area network (SAN) and redundant array of independent disks (RAID)

Server setup

For the demonstration featured in this paper, the following hardware and software configuration was established.

Hardware

The hardware configuration of the server used for these tuning tests was:

- Four 1.2-gigahertz dual-core IBM POWER4+™ chips
  - Two CPUs per POWER4+ chip
  - Total of eight CPUs in the system
- 64-kilobyte L1 I-cache per CPU
- 32-kilobyte L1 D-cache per CPU
- 1.5-megabyte L2 cache per POWER4+ chip
- 32-megabyte L3 cache
- 16-gigabyte main memory
- Two imbedded Wide Ultra3 SCSI I/O controllers
  - Two 146.8-gigabyte disks
  - One 4.7-gigabyte SCSI DVD-ROM drive
- Two PCI-X 2-gigabit per second Fibre Channel adapters
  - Connected to a 16-way switch that is attached to a SAN
  - Two terabytes of disk storage allocated on the SAN for lemans

Software

The following releases of the AIX operating system and SAS software were used for these tuning tests:

- AIX 5L Version 5.3
- SAS Version 9.1.3

Microcode

The first step in setting up the pSeries machine was to run a microcode survey to ensure the system and device firmware were up-to-date. The link to the information needed to run the microcode survey is:

http://www14.software.ibm.com/webapp/set2/mds/fetch?page=mds.html

(**Note:** This link can also be found in the “Resources” section of this paper.)

The inventory scout program was run as root to collect device microcode levels:

```
# invscout
```

On the Web site referenced above, the link **Upload a data file** was used to send the file /var/adm/invscout/lemans.mup (produced by the `invscout` command) to IBM. Within a few seconds, a report was sent back with links to the microcode files and readme files for installing the microcode updates on lemans. In this demonstration, most I/O devices on the system required firmware updates.

AIX

To ensure that AIX was up-to-date, the AIX 5L V5.3 installation CDs were used to migrate lemans from AIX 5L V5.2 to AIX 5L V5.3. This migration went smoothly. Then the latest AIX 5L V5.3 fixes were applied.

SAS

SAS V9.1.3 with service pack 2 was already installed on lemans, and the installation remained intact through the AIX 5L V5.2 to AIX 5L V5.3 migration. In addition, a specific fix from service pack 3 for SAS V9.1.3 was installed that improved AIX memory allocation efficiency for requests from SAS V9.1.3.

SAN and file system

The SAN resources supplied by the SAS R&D Data Center provided three virtual path devices: vpath0, vpath1, and vpath2. All physical disks in the SAN were 10,000 rpm, 146-gigabyte Fibre Channel disks. Multiple physical disks in the SAN were allocated in RAID-5 configurations and accessed through more than one path to implement the vpath0 and vpath1 devices. Several physical disks in the SAN were also allocated in RAID-0 configurations and accessed through multiple paths to implement the vpath2 device. The `datapath query device` command was run to display the SAN configuration shown in Listing 1:

```
# datapath query device

DEV#:   0  DEVICE NAME: vpath0  TYPE: 2145            POLICY:    Optimized
SERIAL: 600507680180809808000000000000002
==========================================================================
Path#      Adapter/Hard Disk          State     Mode     Select     Errors
    0          fscsi0/hdisk4           OPEN   NORMAL    1494403          0
    1          fscsi0/hdisk6           OPEN   NORMAL          0          0
    2          fscsi1/hdisk8           OPEN   NORMAL    1494404          0
    3         fscsi1/hdisk10           OPEN   NORMAL          0          0

DEV#:   1  DEVICE NAME: vpath1  TYPE: 2145            POLICY:    Optimized
SERIAL: 600507680180809808000000000000003
==========================================================================
Path#      Adapter/Hard Disk          State     Mode     Select     Errors
    0          fscsi0/hdisk5           OPEN   NORMAL        201          0
    1          fscsi0/hdisk7           OPEN   NORMAL          0          0
    2          fscsi1/hdisk9           OPEN   NORMAL        189          0
    3         fscsi1/hdisk11           OPEN   NORMAL          0          0

DEV#:   2  DEVICE NAME: vpath2  TYPE: 2145            POLICY:    Optimized
SERIAL: 600507680180809808000000000000008
==========================================================================
Path#      Adapter/Hard Disk          State     Mode     Select     Errors
    0         fscsi0/hdisk12           OPEN   NORMAL          0          0
    1         fscsi0/hdisk13           OPEN   NORMAL    5524671          0
    2         fscsi1/hdisk14           OPEN   NORMAL          0          0
    3         fscsi1/hdisk15           OPEN   NORMAL    5779119          0
```

Listing 1: SAN configuration on lemans

Each hdisk in a vpath represents a connection to the storage allocated in the SAN for the vpath device. By accessing the SAN through vpath devices, the data-path optimizer pseudo-device driver can balance loads across multiple paths.

As with native hdisk devices, vpath devices can be used by AIX to hold logical volumes for file systems. On lemans, hdisk13 and hdisk15 implemented the /dev/raid0lv01 logical volume (mounted over /raid0), and hdisk4 and hdisk8 implemented the /dev/raid5lv01 logical volume (mounted over /raid5). Both volumes had 998.5-gigabyte jfs2 file systems. See Table 1:

File system mount point	Logical volume	hdisk	Adapter
/raid0	/dev/raid0lv01	hdisk13	fscsi0
		hdisk15	fscsi1
/raid5	/dev/raid5lv01	hdisk4	fscsi0
		hdisk8	fscsi1

Table 1: Logical volumes implemented by vpath devices

Data mart setup

Large raw data files and file sets were generated on lemans to simulate an ETL environment that a large retail store chain might implement. These files and file sets were stored in the /raid5 file system. The /raid0 file system was used for **work**, **utility**, and other temporary files.

Initial I/O performance assessment

The upper limit of system performance was determined by using the namefs DIO mounts of the /raid5 and /raid0 file systems to read and write files (via the `dd` command). The pseudo-device /dev/null was specified as the data sink, and the pseudo-device /dev/zero was specified as the data source. The filemon program was invoked to measure performance.

Direct I/O mounts

The direct I/O mounts were created using the following commands:

```
# mkdir /raid0dio /raid5dio
# mount –o dio /raid0 /raid0dio
# mount –o dio /raid5 /raid5dio
```

filemon command

The following `filemon` command was run as a non-root user. (However, the user ID had to be a member of the system group.)

```
$ filemon –dPo filemon.raid0_w –O lf,lv,pv –T 1048576
```

This command line allocated a 1-megabyte trace buffer and pinned it in memory. The output was written to `filemon.raid0_w` and statistics were recorded for the logical file system, logical volume, and physical volume layers. The –d flag told filemon to wait until the `trcon` command was run to start tracing. (For more information on filemon, refer to the “Resources” section of this paper.)

dd command

To drive the data transfer, the `dd` command was used. In addition, the /dev/zero pseudo-device was specified as a zero-latency infinite source of bytes with the value 0, and the /dev/null pseudo-device was used as a zero-latency infinite bit bucket. To ensure that the physical volume layer operated with its maximum transfer size, a 1024-kilobyte blocking factor was used. Such a large blocking factor ensured

the ability to determine the maximum transfer size for hdisk4, hdisk8, hdisk13 and hdisk15 by examining the filemon output. Listing 2 shows the setup for the filemon trace, starting the trace, running the dd command, and then stopping the trace.

```
$ filemon –dPo filemon.raid0_w –O lf,lv,pv –T 1048576
$ trcon; dd if=/dev/zero of=/raid0dio/zeros bs=1024k count=1024; trcstop

$ filemon –dPo filemon.raid0_r –O lf,lv,pv –T 1048576
$ trcon; dd of=/dev/null if=/raid0dio/zeros bs=1024k count=1024; trcstop

$ filemon –dPo filemon.raid5_w –O lf,lv,pv –T 1048576
$ trcon; dd if=/dev/zero of=/raid5dio/zeros bs=1024k count=1024; trcstop

$ filemon –dPo filemon.raid5_r –O lf,lv,pv –T 1048576
$ trcon; dd of=/dev/null if=/raid5dio/zeros bs=1024k count=1024; trcstop
```

Listing 2: filemon trace setup

Results from running dd

The filemon reports were examined, and the following characteristics were determined:

```
Logical Volume     Write Speed       Read Speed
==============     =============     =============
/dev/raid0lv01     118892.7 KB/s     117877.7 KB/s
/dev/raid5lv01     164033.2 KB/s     144576.4 KB/s

hdisk              MAX Transfer Size
==============     =================
hdisk4             256 KB
hdisk8             256 KB
hdisk13            256 KB
hdisk15            256 KB
```

These results showed that the maximum transfer size was 256 kilobytes. Therefore, for optimal efficiency when moving data between the file system cache and the SAN, AIX needs to use 256-kilobyte transfers as often as possible. The following parameters were fine-tuned to increase the occurrence of the 256-kilobyte transfers:

```
Name                            Default value   New value
j2_minPageReadAhead             2               16
j2_nPagesPerWriteBehindCluster  32              128
sync_release_ilock              0               1
```

For more information on these parameters, refer to the “Resources” section of this paper.

Initial PROC SORT performance

Although sometimes hidden behind more complex data manipulation processes, the sorting and merging of data sets is a major aspect of the operation of large data marts. Massive sort jobs, however, can disrupt the performance of multiuser systems when they occupy the file system cache. Tuning a system for efficient sorting and merging, as well as tweaking large sorts to minimize their impact on system resources, covers a significant part of the performance problem.

Setup

In this tuning exercise, a 17-gigabyte compressed data set, from the data mart on lemans, was sorted by a major and minor key using the SAS procedure PROC SORT. The data set contained 175,889,848 store sales records. When uncompressed, the size was 60 gigabytes. With 16 gigabytes of memory installed on lemans, PROC SORT required the creation of a temporary disk-resident SAS utility file. The directory for the location of this file was specified with the option UTILLOC. The utility and work directories were located on /raid0, which was set up for temporary storage. The input data set and the destination for the output data set were on the /raid5 file system. The SORTSIZE option was set to 4 gigabytes, and the MEMSIZE option was set to 8 gigabytes. The filemon program was then run to collect I/O data during the sort. In addition, the topas performance monitor was run in a separate window. For more information on the topas performance monitor, see the "Resources" section of this paper.

Initial adjustments

On the first run, the report showed that the trace had missed a large number of events. To eliminate this problem, the filemon command was run at a higher priority with a larger trace buffer. The command now had to be run as root to use nice to raise its priority:

```
# nice --10 filemon –dPo filemon.raid5_r –O lf,lv,pv –T 16777216
```

The topas monitor also showed that part of the SAS process was paging out due to the demand for a large number of pages by the file system cache. This contention between computational pages (SAS and other programs) and noncomputational pages (the file systems) caused the system to thrash. To protect computational pages from being paged out, the default settings of the following AIX performance parameters were changed:

```
Name             Default value   New Value
maxperm          80              60
strict_maxperm   0(off)          1(on)
```

With the strict_maxperm parameter turned on, the new value of maxperm limited the percentage of real memory that was being used to store noncomputational pages. After making these changes, SAS and other programs on the system occupied 39.2% of real memory without paging. As expected, the filemon output subsequently showed no activity on the page space logical volumes.

First run results

Highlights from the SAS log file and the filemon report are listed below:

The SAS log file:

```
NOTE: The SAS System used:
      real time           33:35.62
      user cpu time       24:43.67
      system cpu time     13:59.67
      Memory                              4212800k
      Page Faults                         738014
      Page Reclaims                       4419392
      Page Swaps                          0
      Voluntary Context Switches          3949910
      Involuntary Context Switches        41236
```

The filemon report:

```
Most Active Files
--------------------------------------------------------------------------------
#MBs       #opns   #rds    #wrs     file
--------------------------------------------------------------------------------
120804.2   2       966434  966434   utB1B2000002.utl
34702.9    1       103426  2117568  sorted_stg_sales_fb.sas7bdat.lck
16557.4    1       1059674 0        sorted_stg_sales.sas7bdat
--------------------------------------------------------------------------------
Most Active Logical Volumes
--------------------------------------------------------------------------------
util  #rblk      #wblk      KB/s     volume            description
--------------------------------------------------------------------------------
0.58  124062200  123704176  61456.9  /dev/raid0lv01    /raid0
0.47  33915688   66071968   24801.3  /dev/raid5lv01    /raid5
--------------------------------------------------------------------------------
Most Active Physical Volumes
--------------------------------------------------------------------------------
util  #rblk    #wblk    KB/s     volume         description
--------------------------------------------------------------------------------
1.00  17056168 32818472 12371.1  /dev/hdisk4    SAN Volume Controller Device
1.00  16859520 33253496 12430.2  /dev/hdisk8    SAN Volume Controller Device
0.99  62029776 62478216 30883.4  /dev/hdisk15   SAN Volume Controller Device
0.99  62032424 61226592 30573.6  /dev/hdisk13   SAN Volume Controller Device
--------------------------------------------------------------------------------
Detailed Physical Volume Stats   (512 byte blocks)
--------------------------------------------------------------------------------
VOLUME: /dev/hdisk4  description: SAN Volume Controller Device
reads:  40543
read sizes  (blks): avg 420.7   min 8   max 512   sdev 132.8
writes: 744815
write sizes (blks): avg  44.1   min 8   max 512   sdev  83.7
--------------------------------------------------------------------------------
VOLUME: /dev/hdisk8  description: SAN Volume Controller Device
reads:  40201
read sizes  (blks): avg 419.4   min 8   max 512   sdev 133.9
writes: 767552
write sizes (blks): avg  43.3   min 8   max 512   sdev  83.4
--------------------------------------------------------------------------------
VOLUME: /dev/hdisk15  description: SAN Volume Controller Device
reads:  126561
read sizes  (blks): avg 490.1   min 8   max 512   sdev  97.3
writes: 122844
write sizes (blks): avg 508.6   min 8   max 512   sdev  41.1
--------------------------------------------------------------------------------
VOLUME: /dev/hdisk13  description: SAN Volume Controller Device
reads:  126470
read sizes  (blks): avg 490.5   min 8   max 512   sdev  96.4
writes: 120319
write sizes (blks): avg 508.9   min 8   max 512   sdev  39.3
```

First run analysis

In this job, a 17-gigabyte compressed data set was sorted by a major and minor key, and the new data set was saved to disk. The input data set sorted_stg_sales.sas7bdat was read by SAS using 16-kilobyte reads. The output data set sorted_stg_sales_fb.sas7bdat was read and written by SAS using 16-kilobyte reads and writes. In addition, the PROC CONTENTS procedure, which describes the structure of the data set rather than the values contained in it, was run against both data sets. The SAS logs reported a page size of 16 kilobytes for both data sets, and showed that the temporary utility files were read and written by SAS at 64-kilobyte page sizes.

An advantage of going through the file system cache was that larger I/O requests were presented to the disk subsystem, meaning that smaller I/O requests had been coalesced into larger requests. For every five I/O requests presented to the file system layer by SAS, approximately two requests were sent to the lower disk subsystem layer.

The information in the filemon report was used to compare the total amount of data moved through the file system level and the disk I/O level. In this case, only 1.3% more data were moved through the file system level, which meant that this job offered little opportunity for reuse of data in the file system cache. This type of job, when run on busy systems, tends to pollute the file system cache with low reuse data and can potentially replace high reuse data.

The sort job used most of the available system memory on lemans but only consumed a little more than one CPU worth of processing power on this 8-way system. The RAID-5 logical disk was used at 43% of its I/O capacity, and the RAID-0 logical disk was used at 21% of its I/O capacity.

PROC SORT performance using direct I/O

The next step was to investigate the performance of the sort job without the use of the file system cache. This was done by accessing /raid0 and /raid5 through direct I/O mounts. It was reasoned that, in a multiuser system, using direct I/O for the sort job would make the file system cache more available to other jobs. An important constraint during this tuning effort was that the run time of the sort job itself needed to remain within an acceptable range.

Second run results

Highlights from the SAS log file and the filemon report are listed below:

The SAS log file:

```
NOTE: The SAS System used:
      real time           52:32.08
      user cpu time       24:41.30
      system cpu time     8:29.05
      Memory                              4212800k
      Page Faults                         154
      Page Reclaims                       4203677
      Page Swaps                          0
      Voluntary Context Switches          9194784
      Involuntary Context Switches        9886
```

The filemon report:

```
Most Active Files
------------------------------------------------------------------------
#MBs       #opns   #rds     #wrs     file
------------------------------------------------------------------------
120804.2   2       966434   966434   utA116000002.utl
34702.9    1       103426   2117568  sorted_stg_sales_fb.sas7bdat.lck
16557.4    1       1059674  0        sorted_stg_sales.sas7bdat
------------------------------------------------------------------------
Most Active Logical Volumes
------------------------------------------------------------------------
util  #rblk      #wblk      KB/s     volume           description
------------------------------------------------------------------------
0.47  123703552  123703552  39242.5  /dev/raid0lv01   /raid0
0.37  37238368   67846032   16668.0  /dev/raid5lv01   /raid5
------------------------------------------------------------------------
Most Active Physical Volumes
------------------------------------------------------------------------
util  #rblk    #wblk     KB/s    volume          description
------------------------------------------------------------------------
0.99  18629632 34148408 8371.4   /dev/hdisk4     SAN Volume Controller Device
0.99  18608736 33697624 8296.6   /dev/hdisk8     SAN Volume Controller Device
0.23  61948800 61872896 19640.0  /dev/hdisk13    SAN Volume Controller Device
0.23  61754752 61830920 19602.6  /dev/hdisk15    SAN Volume Controller Device
------------------------------------------------------------------------
Detailed Physical Volume Stats   (512 byte blocks)
------------------------------------------------------------------------
VOLUME: /dev/hdisk4  description: SAN Volume Controller Device
reads:  596445
read sizes  (blks): avg  31.2 min   8   max  32   sdev 3.5
writes: 1070721
write sizes (blks): avg  31.9 min   8   max 120   sdev 1.6
------------------------------------------------------------------------
VOLUME: /dev/hdisk8  description: SAN Volume Controller Device
reads:  595849
read sizes  (blks): avg  31.2 min   8   max  32   sdev 3.5
writes: 1056589
write sizes (blks): avg  31.9 min   8   max 128   sdev 1.6
------------------------------------------------------------------------
VOLUME: /dev/hdisk13  description: SAN Volume Controller Device
reads:  483975
read sizes  (blks): avg 128.0 min 128   max 128   sdev 0.0
writes: 483397
write sizes (blks): avg 128.0 min   8   max 128   sdev 0.7
------------------------------------------------------------------------
VOLUME: /dev/hdisk15  description: SAN Volume Controller Device
reads:  482459
read sizes  (blks): avg 128.0 min 128   max 128   sdev 0.0
writes: 483070
write sizes (blks): avg 128.0 min   8   max 128   sdev 0.7
```

Second run analysis

Direct I/O decreased the consumption of system CPU time by 53.3%, but it increased run time by 56.4% because of an increase in I/O wait time. There was no read ahead to overlap with I/O startup latency as there was when the file system cache was used. The physical volume report showed that the average block sizes for reads and writes were smaller, resulting in less efficient I/O. There were also more I/O transactions; therefore, total transaction startup latency increased.

Comparison of this job with the previous run (with file system cache) showed that the additional 5.5 minutes of system CPU time, in the first run, was spent moving data between SAS I/O buffers and the file system cache. In this respect, direct I/O is more efficient because the I/O adapters use direct memory

access (DMA) to move data directly to and from SAS I/O buffers. Direct I/O does not need the CPU-driven memory-to-memory copy that moves data between the file system cache and SAS I/O buffers.

As expected, the user CPU time was essentially the same, regardless of whether cached or direct I/O was used. The reason for this is that the processing measured by user CPU time is essentially unchanged by the external I/O method. Even though 19 minutes was added to the sort job's run time, more total work could be done on the system. This is the result of an additional 60% of memory that was freed, and seven other CPUs were still idle and available to do work.

PROC SORT performance using direct I/O (data set PAGESIZE=256K)

With direct I/O, SAS is in direct control of the transfer size of the low-level I/O. The previous analysis pointed out some of the benefits of direct I/O, but it was still necessary to tune SAS to perform more efficient direct I/O transfers. As a result, a second test was run with a data set page size of 256 kilobytes instead of the 16 kilobytes specified in the previous demonstration. To change the data set page size, the program shown in Listing 3 was used:

```
options fullstimer compress=yes;
libname in '/raid5dio/disk3/compress_data';
libname out '/raid5dio/frbart';
data out.sorted_stg_sales_256k (BUFSIZE=256K);
     set in.sorted_stg_sales;
run;
```

Listing 3: Changing data set page size to 256k

Note: If the FULLSTIMER option is turned on, the simple statistics that appear in the SAS log will be greatly expanded.

The sort job was then run with the new input data set and **–bufsize 256K** set on the SAS command line, which caused new data sets to be created with a 256-kilobyte page size.

Third run results

Highlights from the SAS log file and the filemon report are listed below:

The SAS log file:

```
NOTE: The SAS System used:
      real time           38:59.46
      user cpu time       24:12.47
      system cpu time     6:31.93
      Memory                              4214650k
      Page Faults                         65
      Page Reclaims                       2829005
      Page Swaps                          0
      Voluntary Context Switches          6089601
      Involuntary Context Switches        16173
```

The filemon report:

```
Most Active Files
------------------------------------------------------------------------
#MBs       #opns   #rds    #wrs    file
------------------------------------------------------------------------
120804.2   2       966434  966434  ut405A000002.utl
32571.8    1       1       130294  sorted_stg_sales_256k_fb.sas7bdat.lck
16442.0    1       65770   0       sorted_stg_sales_256k.sas7bdat
------------------------------------------------------------------------
Most Active Logical Volumes
------------------------------------------------------------------------
util   #rblk      #wblk      KB/s     volume          description
------------------------------------------------------------------------
0.65   123703552  123703552  52872.9  /dev/raid0lv01  /raid0
0.16   33674088   66706496   21452.1  /dev/raid5lv01  /raid5
------------------------------------------------------------------------
Most Active Physical Volumes
------------------------------------------------------------------------
  util  #rblk    #wblk    KB/s     volume        description
------------------------------------------------------------------------
  1.00  16733376 33425960 10719.5  /dev/hdisk8   SAN Volume Controller Device
  1.00  16940712 33280536 10732.7  /dev/hdisk4   SAN Volume Controller Device
  0.32  61764224 61867464 26421.1  /dev/hdisk15  SAN Volume Controller Device
  0.32  61939328 61836232 26451.8  /dev/hdisk13  SAN Volume Controller Device
------------------------------------------------------------------------
Detailed Physical Volume Stats   (512 byte blocks)
------------------------------------------------------------------------
VOLUME: /dev/hdisk8  description: SAN Volume Controller Device
reads:  32715
read sizes  (blks): avg 511.5   min   8   max 512   sdev 15.4
writes: 65290
write sizes (blks): avg 512.0   min   8   max 512   sdev  4.4
------------------------------------------------------------------------
VOLUME: /dev/hdisk4  description: SAN Volume Controller Device
reads:  33120
read sizes  (blks): avg 511.5   min   8   max 512   sdev 15.0
writes: 65004
write sizes (blks): avg 512.0   min   8   max 512   sdev  3.4
------------------------------------------------------------------------
VOLUME: /dev/hdisk15  description: SAN Volume Controller Device
reads:  482533
read sizes  (blks): avg 128.0   min 128   max 128   sdev  0.0
writes: 483348
write sizes (blks): avg 128.0   min   8   max 128   sdev  0.5
------------------------------------------------------------------------
VOLUME: /dev/hdisk13  description: SAN Volume Controller Device
reads:  483901
read sizes  (blks): avg 128.0   min 128   max 128   sdev  0.0
writes: 483104
write sizes (blks): avg 128.0   min   8   max 128   sdev  0.5
```

Third run analysis

When compared to the previous run, the use of a 256-kilobyte page size for the input and output data sets reduced run time by 25.8% and system CPU time by 23.0%. This improvement brought the run time closer to the run time achieved when using the file system cache. Although an additional 5.5 minutes was required, 9.6 gigabytes (out of a total of 16 gigabytes) of main memory was freed. These results suggested that further investigating changes to the page size of the utility file might yield even better results.

PROC SORT performance using direct I/O (data set and utility file PAGESIZE=256K)

It was unclear whether there was a way to specify the page size of the utility file that PROC SORT uses. To test this uncertainty, the option BUFSIZE=256K was tried with PROC SORT. This generated a syntax error message in the log file, although PAGESIZE was one of the options listed in the error message after the words “expecting one of the following.” PAGESIZE=256K was tried, and it worked.

Fourth run results

Highlights from the SAS log file and the filemon report are listed below:

The SAS log file:

```
NOTE: The SAS System used:
      real time           33:59.30
      user cpu time       24:01.52
      system cpu time     5:22.80
      Memory                              4214650k
      Page Faults                         101
      Page Reclaims                       4208930
      Page Swaps                          0
      Voluntary Context Switches          1735701
      Involuntary Context Switches        6816
```

The filemon report:

```
Most Active Files
  ------------------------------------------------------------------------
  #MBs       #opns   #rds    #wrs     file
  ------------------------------------------------------------------------
  120804.5   2       241609  241609   utB14A000002.utl
  32571.8    1       1       130294   sorted_stg_sales_256k_fb.sas7bdat.lck
  16442.0    1       65770   0        sorted_stg_sales_256k.sas7bdat
  ------------------------------------------------------------------------
Most Active Logical Volumes
  ------------------------------------------------------------------------
  util  #rblk      #wblk      KB/s     volume           description
  ------------------------------------------------------------------------
  0.50  123703808  123703808  60654.4  /dev/raid0lv01   /raid0
  0.20  33673816   66706496   24609.2  /dev/raid5lv01   /raid5
  ------------------------------------------------------------------------
Most Active Physical Volumes
  ------------------------------------------------------------------------
  util  #rblk     #wblk     KB/s     volume        description
  ------------------------------------------------------------------------
  1.00  16836632  33490464  12338.2  /dev/hdisk4   SAN Volume Controller Device
  1.00  16837184  33216032  12271.0  /dev/hdisk8   SAN Volume Controller Device
  0.25  61881856  61801624  30322.2  /dev/hdisk15  SAN Volume Controller Device
  0.25  61821952  61902472  30332.3  /dev/hdisk13  SAN Volume Controller Device
  ------------------------------------------------------------------------
Detailed Physical Volume Stats   (512 byte blocks)
  ------------------------------------------------------------------------
  VOLUME: /dev/hdisk4  description: SAN Volume Controller Device
  reads:  32899
  read sizes  (blks): avg 511.8   min   8   max 512   sdev  9.2
  writes: 65415
  write sizes (blks): avg 512.0   min   8   max 512   sdev  3.9
  ------------------------------------------------------------------------
  VOLUME: /dev/hdisk8  description: SAN Volume Controller Device
  reads:  32902
  read sizes  (blks): avg 511.7   min   8   max 512   sdev 10.7
  writes: 64879
  write sizes (blks): avg 512.0   min   8   max 512   sdev  4.0
  ------------------------------------------------------------------------
  VOLUME: /dev/hdisk15  description: SAN Volume Controller Device
  reads:  120863
  read sizes  (blks): avg 512.0   min 512   max 512   sdev  0.0
  writes: 120725
  write sizes (blks): avg 511.9   min   8   max 512   sdev  6.3
  ------------------------------------------------------------------------
  VOLUME: /dev/hdisk13  description: SAN Volume Controller Device
  reads:  120746
  read sizes  (blks): avg 512.0   min 512   max 512   sdev  0.0
  writes: 120920
  write sizes (blks): avg 511.9   min   8   max 512   sdev  6.0
```

Fourth run analysis

Direct I/O, coupled with 256-kilobyte page sizes for both the utility file and the data sets, improved run time so that it was only 1.2% slower than the first run (which used file system cache, 16-kilobyte page size data sets, and a 64-kilobyte page size utility file). CPU time was also reduced by 62%. Detailed physical volume statistics showed that the size of almost every physical volume transfer was 256

kilobytes. Compared to the previous run, adding PAGESIZE=256K reduced run time by 12.8% and CPU time by 17.6%.

PROC SORT performance using direct I/O (data set and utility file PAGESIZE=256K, BUFNO=16)

One advantage of file system caching that is not available with direct I/O is that the AIX virtual memory manager (VMM) uses **read ahead** and **write behind** to pipeline I/O for reduced latency. To compensate for loss of the AIX VMM's pipelining when using direct I/O, the SAS BUFNO option was set to greater than 1 to enable SAS to pipeline I/O requests. For the final experiment, the previous job was run again with –bufno 16 added to the command line.

Final run results

Highlights from the SAS log file are listed below:

The SAS log file:

```
NOTE: The SAS System used:
      real time           31:32.07
      user cpu time       23:59.68
      system cpu time     4:45.56
      Memory                              4218234k
      Page Faults                         79
      Page Reclaims                       4212107
      Page Swaps                          0
      Voluntary Context Switches          1675767
      Involuntary Context Switches        3942
```

Final run analysis

Comparing the memory needed in the previous and current runs showed that SAS used 14 x 256 kilobytes more memory. It was speculated that, minimally, SAS uses two 256-kilobyte I/O buffers, adding 14 more to meet the request for 16. When compared to the previous run, the addition of –bufno 16 reduced run time by 7.2% and CPU time by 11.5%.

One more run was done with –bufno 64 and compared to the –bufno 16 run. The –bufno 64 run took an additional 16 seconds off the run time, 20 seconds off CPU time, and increased SAS memory requirements by 48 x 256 kilobytes. These results suggest future work is needed to plot real time and system CPU time against the number of SAS I/O buffers.

Compared to the very first run, which used the file system cache, this final run completed in 6% less time, system CPU time was reduced by 68%, user CPU time was reduced by 3%, and file system cache resources were no longer consumed by the sort job.

Summary

In the study reviewed in this paper, AIX parameters and SAS options were tuned to reduce the impact of a large sort job on system resources for the purposes of minimizing the job's performance impact when run in a multiuser environment. However, the tuning effort occurred in a controlled single-user environment that was used to facilitate managed performance experiments. Ultimately, the changes suggested in this document need to be validated in an actual multiuser scenario and, as such, might not yield the results obtained in this set of tests.

Before attempting performance tuning, it was necessary to ensure that the p650 firmware and AIX 5L V5.3 were at the recommended maintenance levels, and that the SAS installation was functioning properly. Basic information about the hardware to provide a framework of understanding of the I/O subsystem was presented. Performance characteristics of the SAN were determined and used for initial performance tuning of the operating system. It was discovered that, with AIX 5L V5.3, most of the default I/O and VMM tuning parameters were already set to reasonable values for enterprise-level work, which had not been the case with pre-AIX 5L V5.3 systems.

The filemon utility (which is based on the AIX trace facility), the topas monitor, and the output of the fullstimer option in the SAS log provided data for the tuning decisions. A large sort job was selected as a representative workload, and the AIX tuning parameters maxperm and strict_maxperm were adjusted to keep the system from paging out the SAS process. Direct I/O was selected as the first tuning step, because the uncompressed size of the input data set was larger than the available memory, and analysis of the first job runs showed poor file system cache reuse.

Initially, the use of direct I/O freed up the file system cache, but increased run time. The rest of the tuning effort focused on compensating for the loss of benefits of using the file system cache. First, the input data set page size was changed from 16 to 256 kilobytes, and the output data set page size was specified as 256 kilobytes. This brought the run time back down to an acceptable level, further reduced system CPU time, and continued to provide the benefit of freeing up the file system cache. The PAGESIZE option of PROC SORT specified the page size that SAS used for the utility file, and a utility file page size of 256 kilobytes was used, further reducing run time and system CPU time. Finally, increasing the number of I/O buffers for SAS reduced run time below that of the first run and provided a further reduction in the use of system CPU time.

The AIX 5L operating system and the SAS application set are instrumented to provide tools and facilities for producing performance reports. Both offerings deliver tuning parameters that can be changed to improve both the run time and the efficient utilization of system resources. A successful tuning effort requires up-to-date hardware and software, incremental parametric changes, a data-driven approach, careful analysis of performance data, and an accurate conceptual model of the system I/O.

Resources

These Web sites provide useful materials to supplement the information contained within this document.

- IBM Techdocs (the Technical Sales Library)

  ibm.com/support/techdocs/
  - Hints and Tips for Running SAS on an IBM eServer™ pSeries Server and AIX 5L (TD102517)
  - Tune AIX 5L for the SAS 9 System (TD102515)

- AIX 5L Performance guides:

  publib.boulder.ibm.com/infocenter/pseries/index.jsp

  Navigate **AIX documentation** > **AIX PDFs** > then, under **Performance**
  - AIX 5L V5.3 Performance Management Guide
    in the right pane, click **Performance Management Guide**
  - AIX 5L V5.3 Performance Tools Guide and Reference
    in the right pane, click **Performance Tools Guide and Reference**
  - Performance Toolbox Version 2 and 3 Guide and Reference
    in the right pane, click **Performance Toolbox Version 2 and 3 Guide**

- Microcode survey information

  http://www14.software.ibm.com/webapp/set2/mds/fetch?page=mds.html

- AIX commands and tools information:

  http://publib.boulder.ibm.com/infocenter/pseries/v5r3/index.jsp

- Navigate **AIX documentation** > **Commands reference** > **Alphabetical list of commands** >
  - For usage of **filemon**, continue navigating > **f** > **filemon**
  - For usage of the tunable input and output parameters, continue navigating > **i** > **ioo**
  - For usage of the topas performance monitor, continue navigating > **t** > **topas**
- SAS Home page

  sas.com

- SAS and IBM white papers

  sas.com/partners/directory/ibm/papers.html
  - Configuration Options using IBM SSA Storage for SAS
  - Maximizing Performance on IBM Enterprise Storage Servers®
  - High Availability Storage on IBM Enterprise Storage Servers
  - Managing SAS Storage on IBM Enterprise Storage Servers

- A Practical Approach to Solving Performance Problems with the SAS System (October 2001)

  support.sas.com/rnd/papers/sugi27/SolvingPerformance.pdf

- IBM eServer p5 Information Center

  publib.boulder.ibm.com/infocenter/pseries/index.jsp

- IBM Publications Center

  www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi?CTY=US

- IBM Redbooks™

  www.redbooks.ibm.com/
  - AIX 5L Practical Performance Tools and Tuning Guide (SG24-6478)

About the author

Frank Bartucca

IBM Systems and Technology Group

Raleigh, North Carolina

Mr. Bartucca is a senior engineer and scientist with IBM eServer Solutions Enablement in the IBM Systems and Technology Group. He has been working with AIX and IBM RISC technologies since 1984. In his current capacity, as a pSeries technical consultant, he provides technical support to software developers and solution providers who are developing software products to run on the AIX operating system.

Trademarks and special notices

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both: IBM, the IBM logo, eServer, pSeries, AIX, AIX 5L, Redbooks, Enterprise Storage Server, and ibm.com.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

Information is provided "AS IS" without warranty of any kind.

All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

Information concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products. Questions on the capability of non-IBM products should be addressed to the supplier of those products.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.