Tuning Considerations for Version 6 of the SAS System on MVS

Isolde Schell-Haungs, SAS Institute GmbH

ABSTRACT

Performance and resource utilization of Version 6 of the SAS System can be affected by the use of tuning parameters. The impact and tradeoffs of various strategies will be examined with respect to small, medium and large jobs. The paper also discusses performance and resource consumption aspects of SAS application design and programming.

This paper is based on two SUGI 16 papers, read in NEW Orleans at the 16th Annual Conference, February 17-20, 1991:


Daniel J. Squillace, SAS Institute Inc., Cary, NC: Tuning Your SAS Application under MVS

INTRODUCTION

Usually, optimization means balancing the tradeoff between speed and size. Where and how this balance is done is dependent on many things, such as factors associated with the user's installation, SAS job size, contention and so on. There are a number of techniques you can use to optimize the performance of your SAS programs under MVS. Some of these techniques such as bundling are site decisions.

Some techniques are host specific aspects that you should be aware of, while others, such as using the performance statistics reported in the SAS log to tune your SAS program apply to any operating system. In any case by using a combination of these techniques as appropriate for your particular programming situation you can optimize the performance of the SAS system.

In this paper following topics are covered:
- Bundled Version
- SAS System Options to control the output of performance statistics, to control the memory usage and to control sorting
- The structure of SAS data libraries under MVS and the options that can help you to optimize performance and disk space.
- Efficient programming of SAS applications

The focus of the paper is on ways you can optimize Release 6.06 of the SAS System under MVS. But whenever convenient it is mentioned what will be changed in Release 6.07 and at the end there is a short summary of 6.07 improvements.
**BUNDLED VERSIONS**

Bundling of modules in general eliminates dead space between modules and saves the overhead of loading each module.

There are four bundle configurations for base SAS Software:

- MVS/XA, MVS/ESA and ELPA/LPA
  Entry: SASXAL
- MVS/370 and LPA
  Entry: SAS370L
- MVS/XA, MVS/ESA and non-ELPA/LPA
  Entry: SASXAL
- MVS/370 and non-LPA
  Entry: SAS370

Note that the code is identical among the four versions; the only difference is in the packaging.

The ENTRY parameter of the JCL cataloged procedure or the TSO CLIST determines which configuration is used. If the Link Pack Area (LPA) / Extended Link Pack Area (ELPA) version is used, only one copy of base SAS Software is loaded. If the appropriate non-LPA or non-ELPA/LPA version is used, it is loaded in each SAS user address space and saves valuable ELPA/LPA space, but it can cause a large increase in working size and can possibly place a heavy burden on the paging subsystem.

So ELPA/LPA installation is highly recommended, if there is a significant number of interactive users at an installation.

In addition, some other parts of the SAS System, for example DATA step compiler and execution components, or full screen and SCL (Screen Control Language) code also have bundles that may be installed into ELPA/LPA so you save additionally about 1.4 MB in each SAS user address space.

Instructions for selecting a bundled configuration of the SAS System and for installing the bundled modules of either configuration MVS/370 or MVS/XA and MVS/ESA are documented in Customization Instructions for Base SAS Software under MVS, Release 6.06, chapter "Customizing the Base SAS Software Product".

For the installation and maintenance of Supervisor and Product Bundles in ELPA/LPA under MVS/XA or MVS/ESA, please follow the recommendations in the SUGI 16 paper from Charles A.Jacobs III, SAS Institute Inc, Cary, NC:

*Installing and Maintaining Supervisor and Product Bundles in ELPA/LPA for MVS/XA and MVS/ESA Sites*

**SAS SYSTEM OPTIONS**

SAS System Options are instructions that affect the entire SAS session and control the way the SAS System performs operations. The main reason system options are used is either to streamline the SAS System or to restrict use of some of the options by the user.

There are three types of SAS system options:

- session system options, which can be changed during the SAS job or session
- configuration system options, which must be specified at invocation
- host system options, which are specific to the environment in which the SAS System is running.

System options help optimize SAS data libraries under MVS, sorting procedures, and memory management. You can determine which SAS system options are in effect by using the OPTIONS procedure, which lists the three types.

The combination of the STATS, FULLSTATS and STIMER options causes the SAS System to report CPU time, elapsed time, and EXCP count after each SAS step in an expanded format.

If the MEMRPT option is on, task memory and total memory are additionally reported in the SAS log.

Here is an example of a SAS log listing that shows performance statistics for a short SAS session:

```
NOTE: Copyright(c) 1989 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) Proprietary Software Release 6.xx Licensed to SAS Institute Inc., Site XXXXXXXXX
NOTE: Running on IBM Model 3090 Serial Number XXXXXXX.

NOTE: SAS system options specified are:
NEWS='MVSHOST.SAS.NEWS (N6xx), ALTLOG=OUT

1 NOTE: The installation phase used 0.30 CPU seconds and 2900K.
1 options fullstats;
2 run;
3
4 filename inn '.misc.test(test1)' shr;
5 data x;
6 infile inn;
7 input;
8 run;

2 NOTE: The infile INN is:
Dsname=userid.MISC.TEST (TEST1)
Unit=3380,Volume=XXX101,
Disp=SHR,Blksize=3120,
Lrecl=80,Recfm=FB

NOTE: 16 records were read from the infile INN.
NOTE: The data set WORK.X has 16 observations and 0 variables.
NOTE: The DATA statement used the following resources:
CPU time - 00:00:00.09
Elapsed time - 00:00:01.65
EXCP count - 140
TASK memory - 862K
(34K data, 828K program)
Total memory - 3577K
(1464K data, 2113K program)

3 NOTE: The SAS session used 0.54 CPU seconds and 3601K.
NOTE: SAS Institute Inc., SAS Campus Drive, Cary, NC, USA

The performance statistics fall into three categories:
1. initialization statistics
2. task statistics
3. session totals

When you are optimizing the performance of your SAS programs, the expanded task statistic should be used to provide the information you need to determine the performance effects caused by modifications to the SAS program.

This statistics include the
- CPU time, which is the time required for the CPU to perform the task

```
- elapsed time, which gives the wall clock time required for the task to be completed
- EXCP count, which stands for execute channel program count and counts the number of I/O operations
- task memory, which reports the memory used by the task
- total memory, which reports the total memory used by the SAS System after the task has been completed.

To interpret the performance statistics, you must remember that they relate to three critical computer resources: CPU time, I/O, and memory. Under almost all circumstances, reduction in the use of any of these three resources results in better throughput of a particular job and, in general, results in a reduction in the elapsed time. The statistics that you are probably most interested in are elapsed time and CPU time. But, because the elapsed time represents the actual wall clock time, it is heavily dependent on the current load and the capacity of the system. So, as a general rule, it is not a good idea to use elapsed time as the only criterion for measuring the efficiency of your program. The CPU time statistic can tell you how much time was actually spent on your own job. Also, most data centers use CPU time to calculate data processing charges.

Another criterion for efficiency is EXCP count. The higher the number, the more system resources are being used. So, I/O optimization is the most important area to concentrate on when optimizing performance. In general, there is a tradeoff between conserving memory and enhancing CPU and I/O performance. With ample memory, you can use large buffers that will usually reduce EXCP count and CPU time.

Also to help determine the systemwide SAS usage profile, the Institute has developed an SMF exit. The SMF exit records as user-written SMF records the system resource utilization of individual SAS steps in an MVS environment. The ability to record SMF records needs to be enabled at system installation time, and it is documented in the installation guide.

**MEMORY USAGE**

SAS system options enable you to limit the amount of memory used by the SAS System or to decrease memory fragmentation so you can effectively manage memory resources used by the SAS System. The following SAS system options are used to constrain memory requirements placed on the host by the SAS System.

The MEMSIZE= option specifies a limit on the total amount of memory the SAS System can use. Whenever acquiring new memory resources causes the SAS System to exceed the value of the MEMSIZE= option, the system tries to free up resources to satisfy the request. The value of the MEMSIZE= option applies to memory above and below the 16 MB line and it can be changed at any time.

When the MINSTG system option is set, the SAS System keeps its total memory usage as low as possible at all times. This means that at all DATA- and PROC-step-boundaries, the SAS System frees all unused memory regions and unloads all modules that are not being used.
action keeps total memory at a minimum, but significantly increases CPU time. Therefore this is a special MVS/370 option, where you have more severe memory restraints.

To decrease memory fragmentation, the SAS System is able to obtain large blocks of memory from the operating system, out of which smaller memory requests are satisfied. This is called superblocking which not only reduces fragmentation but also reduces the number of system GETMAIN calls that are issued. The types of memory managed by superblocking are:

- I/O memory below 16 MB
- non-I/O memory above 16 MB
- permanent memory, which can be used for the duration of the SAS session
- temporary memory, which is reusable for another SAS task
- SAS System portable supervisor memory
- SAS System memory management control memory

Memory is acquired from MVS and is maintained in system control areas. The acquisition of memory is made by primary allocation, called initial storage allocation (ISA) and secondary allocation, called overflow storage allocation (OSA). There are eleven SAS System options to set default values for superblocking at SAS invocation. Please refer to Chapter 17, "Optimizing Performance", in the SAS Companion for the MVS Environment, for more details.

SAS System Options That Control Sorting

There are new SAS options that enable the user to take advantage of more recent host sort utility programs and other host features such as MVS/XA and MVS/ESA 31-bit mode. The SORT procedure can access either the SAS sort utility, which can be provided as part of the SAS System, or a host sort utility, which can be provided by either IBM or another software company. The SAS system option SORTPGM= specifies which sort utility is to be invoked. You can specify SORTPGM=SAS, SORTPGM=HOST or SORTPGM=BEST. The name of the host sort utility is given by the SORTNAME= option. SORTPGM=BEST specifies that the SYS System is to choose either the SAS sort utility or the host sort utility, depending on the data to be sorted. The switch from the internal SAS sort utility is made automatically at 500 observations.

There will be a SORTCUTP option in Release 6.07 to control the point at which the host sort is selected instead of the SAS sort. SORTCUTP will be expressed in data volume rather than number of observations. The default for MVS/XA and MVS/ESA will be around 4 MB, for MVS/370 probably 1 MB.

The SORT31PL option specifies that a 31-bit extended parameter list be used to invoke the host sort utility; otherwise, a 24-bit parameter list is used. On MVS/370 systems this makes little difference; however, using a 31-bit parameter list on MVS/XA or MVS/ESA systems can boost performance. If SORT31PL is set, the options SORTEQOP and SORTOPTS should also be
specified to maximize performance.

This is because sorts that currently support a 31-bit parameter list also support the EQUALS option - that means observations with duplicate keys are sorted in the original order - and the OPTIONS statement. SORTSUMF specifies that the host sort supports the SUM FIELD=NONE control statement. Please refer to Chapter 17 in the SAS Companion for the MVS Environment for more details.

SAS DATA LIBRARIES UNDER MVS

A review of Version 6 data library characteristics will help in understanding the I/O recommendation. Version 6 implements Fixed Block Architecture on Count Key Data devices. A SAS data library is a physical sequential data set with record format fixed standard (RECFM=FS), where LRECL=BLKSIZE. A physical block must be a multiple of 512 bytes. In a Version 6 library observations are stored in pages, whereby the pagesize is a multiple of the defined blocksize. That means, a page is a logical entity stored in one or more blocks, a page is the unit of allocation now and the pagesize is not limited by the tracksize of a special device. Compared to a Version 5 data library, we are no longer restricted to a maximum observation length of 32 kB.

There are four options that can help you to optimize performance and make efficient use of disk space for SAS data libraries. Each option can be specified as SAS system option or as a data set option.

The BLKSIZE= option defines the default blocksize of data libraries. You should consider the tradeoffs, when choosing a blocksize. If your members in a library have a large number of large observations choose the largest optimum blocksize that your device supports. For example: 23040 bytes on a 3380 device. If your members have a small number of small observations, or if the size and number of observations are widely varied choose a smaller blocksize. For example: 6144 bytes.

One last point to this: Larger blocksize result in more efficient space utilization on CKD devices. The half-track block is 7% more space efficient than the 6 k block.

The BUFSIZE= option is used to set the page buffer size for a SAS data set. That means the pagesize may vary at the data set level within the library.

The BUFNO= option specifies how many page buffers are to be allocated for an open SAS data set.

Some tradeoffs need to be taken into consideration. More data is transferred per I/O operation with large pages; therefore the EXCP count will be lower. The disadvantage is that more memory is required for large pages, which is reflected in the task and total memory statistics. Additional buffers can mean a lower EXCP count when accessing pages more than once, such as scrolling backwards in FSEDIT or processing very small data sets. They can also substantially reduce the elapsed time for a SAS job that processes mostly sequential data.
With the COMPRESS= option you can set a new, optional observation format that uses variable-length observations. In general, a SAS data set with compressed observations requires less mass storage than the same data set with uncompressed observations. On the other hand, observation compression and the additional bookkeeping needed to manage the reuse of deleted observation space requires more CPU time.

As a result of all above points we recommend small pages for SAS libraries with catalogs and small datasets, for example: 6144 bytes. For larger data sets we recommend larger pages, for example: half track, that means 23040 bytes on a 3380 device. Whenever possible, put catalogs and SAS data sets in separate libraries. If this is not possible, choose the smaller page size because catalogs take typically 1 kB.

As you can imagine, BUFNO * BUFSIZE is an important factor because BUFNO buffers are transferred in a single I/O operation. Therefore, the amount of data transferred in a single I/O is BUFNO * BUFSIZE bytes. But remember, you need BUFNO * BUFSIZE virtual memory for every open data set. Even if virtual memory is not a problem, it is not worth transferring more than 3 tracks in a single I/O because this does not yield significant performance improvements.

APPLICATION TUNING

The SAS System is powerful, flexible, and easy to program. Its design maximizes the computing power available to you while minimizing your workload in programming. At the same time, the design of your SAS application can produce the correct results, while consuming far more resources than necessary.

Below are some short tips for efficient SAS programming:

- Read and write data selectively; that means the fewer variables and observations you process the fewer I/O operations the system must do.

- Minimize the variable lengths and carry only the variables you need.

- Take advantage of SAS procedures. Different procedures can perform similar tasks, but each procedure can perform more efficiently in specific circumstances.

- Minimize sorting. Rearranging data is one of the most common operations in the SAS System and knowing how and when to sort makes a big difference in the CPU time consumed in a program. Release 6.07 will maintain the sort order information with data sets and bypass unnecessary sorts.

- Create indexes for SAS data sets when the data set is relatively large, when the values within the data set are not frequently updated, and when the data set is frequently subset by values of the indexed variable, whereby less than one-third of the observations are selected.

- a WHERE clause can be more efficient than a subsetting IF. WHERE will use at most one index if available.
When possible, use Screen Control Language instead of SUBMIT blocks.

You can find many more tips like this to improve the efficiency of your SAS application in following publication: SAS Programming Tips: A Guide to Efficient SAS Programming.

FUTURE

Beside many new features and enhancements Release 6.07 is a maintenance release with more than 200 zaps installed, so not much maintenance should be necessary. Additionally there are many internal efficiencies and performance improvements.

For example:

- The base engine is significantly improved.

- Format and informat code generation is significantly improved.

- The problem of opening multiple utility data sets for code generation has been solved.

- The scanner and parser use host-specific features now to improve performance.

- There will be a new file format for decompressed data sets resulting in faster compression.

- Index creation will be twice as fast and index tables will be about 25% smaller in size.

- As already mentioned, unnecessary sorts will be bypassed.

- MVS/ESA is exploited by using hiperspace for work data sets.

- Support for PDSEs is also in progress. This may give additional performance benefits over PDSs.

- Release 6.06 has already the possibility to use vector facility instructions, if your system has the IBM 3090 vector facility installed. In 6.07 additional vector support is incorporated. Candidate procedures are REG, MEANS, SUMMARY, SYSLIN and IML. Preliminary data from an experimental version of the MEANS procedure indicate, that you can receive a ten to fifty percent performance improvement depending on the number of variables and observations you process.

- For those sites that have the IBM’s Engineering and Scientific Subroutine Library (ESSL), many routines can be transparently used by SAS/IML software after local link editing of one image.

CONCLUSION

As you have seen, optimization is a balance between speed and size. In conclusion, Version 6 SAS software contains many features that can be used to tune the SAS System in different environments. Some of these features are bundling and SAS system options that help optimization, memory usage, SAS data libraries under MVS, sorting considerations, and other methods of optimization.
REFERENCES


SUGI 16 paper, NEW Orleans, February 17-20, 1991:


Daniel J.Squillace, SAS Institute Inc., Cary, NC: Tuning Your SAS Application under MVS


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

IBM is a registered trademark and MVS/XA and MVS/ESA are trademarks of IBM Corp., Armonk, NY.