

Optimizing SAS® Software on Windows NT®: A Guide to Performance Tuning Your Applications Server

Peter Parker, US Dept. of Commerce, Washington, DC
peter_parker@ita.doc.gov

Abstract

Optimizing SAS software from the perspective of an applications programmer requires writing efficient and clever code. However, here I'll re-examine my computer system with a systems administrator's eye. I'll be more concerned with the speed and the efficiency of the hardware and the operating system, Microsoft Windows NT 4.0, as it runs SAS software.

Optimization is making the computer system run faster by minimizing bottlenecks. Running SAS software to manipulate large data sets and to crunch vast amounts of numbers puts a significant load on a computer. Any system improvements would produce faster query and report turn-around, creating a more efficient office. This paper will report the results of tweaking and testing my NT box with the following bottlenecks in mind:

- ✓ SAS sort size
- ✓ Hard drive efficiency
- ✓ System memory (and memory paging files)
- ✓ CPU (using multiple processors)
- ✓ Cache (software and hardware)

The Problem

Optimization is making the computer system, in this case, Windows NT, perform faster by minimizing bottlenecks. Bottlenecks are processes, software or hardware, which don't keep up with other parts of the computer. These impediments could be hardware, hard drives or memory, or software, poorly written code. One can not remove all bottlenecks. By definition, a bottleneck is the slowest process in the computer. Something will always be ranked in last place. If one doubles the memory, possibly removing the memory bottleneck, then the next slowest process, perhaps the hard drive, would become the bottleneck. Using another metaphor, if you're walking from one side of the room to the other and with each step, you cover exactly half of the distance between you and the far wall, you'll never reach that wall. But you'll get really close to it. The same is true for bottlenecks. You'll never remove all bottlenecks, but you'll get really close to it.

What's true optimization? In theory, it's removing significant bottlenecks, in order to run the computer at its fastest speed. A racecar can not achieve its full potential if its tires are unable to

withstand speeds of 100 miles per hour. Once its tires are upgraded, the next bottleneck would have to be worked on. A racecar exists for the sake of speed alone, and a true optimization for a racecar would entail constant mechanical enhancements. Yet speed alone is not the most relevant factor for most of us, in computing and in crunching numbers. Our goal shouldn't be the never-ending quest for **true optimization**, but rather for **practical optimization** (my term) to get the job done. If a computer program takes six hours to run and it crashes due to a glitch after five hours, then you may have to restart and lose a day. However if you optimize the program so that it finishes in one hour, then you'll have plenty of leeway, in case of program glitches. That would be **practical optimization**. However, suppose you can quadruple the CPUs and double the memory, and instead of finishing the job in an hour, you finish in thirty minutes, saving thirty minutes. The system would be more optimized, at a considerably higher cost, but it's not necessarily **practical optimization**. After you're able to bring wall clock time down to an hour, any further optimization may only have slight value.

Economists refer to this process of **practical optimization** as diminishing marginal returns. If one has ten shovels and only a few laborers, then adding more laborers would optimize the hole digging process. After hiring more than ten laborers, you'll have more diggers than shovels. You'll still be able to dig holes faster, since the diggers

can take breaks and pass the shovels on to freshly rested diggers. However, each additional laborer would not be as efficient as the first ten, because he'll only be working part time. After a certain point, you'll have even fewer holes being dug, because the laborers would be getting into each other's way.

Exploring diminishing marginal returns is one of the points of my paper. The general American approach is that "more is better". More memory, more hard drive space, and more cache is always better. Having Western European and American joke immunity, I'm reminded of the story about the international students writing term papers on elephants. The British's paper was titled "Hunting Elephants," the Frenchman's paper was titled "The Love Life of the Elephant" and the German's paper is titled "An Annotated Bibliography on Elephants, Volume I." The American's paper was titled "Breeding Bigger and Better Elephants." My paper is on how to really breed bigger and better SAS applications on a Windows NT box.

The Solution

To test my ideas, I put a load on SAS software applications to see how my Windows NT box would respond. I created an application that is both CPU and I/O intensive to represent a general real world case, rather than run an CPU intensive case, followed by an I/O intensive case, since one after the other isn't realistic. Also, I'm more concerned with the concept

of wall clock time. I'm less concerned with how long the CPU and I/O processing times are, taken individually but with how long the entire job takes. Once I start an application, I want to know the bottom line of when it'll finish. Knowing the individual processing times is useful for eventually fine-tuning the Windows NT box. Here I'm interested on how particular tweaking, such as adding more memory, will affect how soon a total SAS software job finishes executing.

In order to explain my findings in an intuitive way, I've decided to use the Internet web page format of FAQ. As a computer problem solver, I think in terms of problems and their solutions. When I search Internet knowledge bases, I often search for a question similar to the one I have.

√ **FAQ # 1-** What's a FAQ?

Answer- That's one. (Frequently Asked Question)

√ **FAQ # 2-** If one CPU is good, is two or more better?

Answer- No. At least for running a single SAS software job, it's not. The SAS system is single-threaded software. A thread is a list of instructions to perform a certain task. Multiple-threaded software can perform several tasks simultaneously if the resources are available. Only by running multiple applications such as multiple SAS jobs at the same time could SAS software run faster

on a multiple processor system. Each SAS job would not run faster, but they wouldn't have to share one processor among them.

Windows NT uses symmetrical multiprocessing where the system can take advantage of multi-threaded applications and load them among the different processors, trying to balance the computational load rather than having certain processors always do certain type of processes. Running multiple single-threaded applications could also use these multiple processors.

√ **FAQ # 3-** If one hard drive is good, are more hard drives better?

Answer- Most servers have multiple drives in order to hold more data and to increase the speed of accessing data. Most configurations have the drives set up as RAID's (Redundant Array of Independent Drives) where these multiple drives act as one drive. Raid level 5 is probably the most popular setup, where one can have both speed and fault tolerance. There are many pros and cons to having a RAID set up for processing SAS software programs.

RAID Drives- To Have and To Have Not

Pros:	Cons:
<ol style="list-style-type: none"> 1. Speed- data I/O is fast since the data is split among multiple disks. 2. Fault tolerance- if a disk goes bad, the data can be recreated from information on the other disks, if using certain RAID levels (rather than recreating the data from a slow medium like DAT tape) 3. Hot swappable- a hard disk can be replaced without even turning off the server(see # 2, fault tolerance) 	<ol style="list-style-type: none"> 1. Expensive- a RAID setup would be a major part of the cost of a server. 2. Complications- A new element of complexity has been added. Now the system administrator needs to be trained to handle subtle RAID problems. 3. Fragility- If an NT server crashes, sometimes that may also crash the sensitive RAID drives. Then you'll have to rebuild the data.

Overall, if one's SAS applications are mostly CPU intensive, investing in hard drives won't buy you much speed. However, if one has a lot of data, mass storage could be a major bottleneck. The optimal hard drive set up would be ultra-wide SCSI RAID, which will data transfer rate at an incredible 80 MB per second. Go with RAID level 0, if one needs only speed, taking full advantage of the reading and writing to multiple disks simultaneously. If fault tolerance is also crucial, slow it down with a RAID level 5, which allow data recoveries on the fly.

However, there are a few caveats with going with the high powered RAID solution. While there are major pluses to be had, the negatives are significant. With new solutions also come new problems. RAIDS can be temperamental and also require new expertise. They are also more sensitive to system crashes. Like Windows NT being a more robust operating system than Windows98, it also requires a

systems operator with higher level skills. For instance, while RAID level 5 is excellent for recovering from a bad disk drive, what do you do when two disk drives go bad? What if one disk drive knocks out two others? RAID systems are like new cars with computerized components in the engine. These engines run much more efficiently, but one needs more specialized equipment and knowledge to fix them. There's a trade-off between efficiency and complexity. With more efficiency can come more complexity. One shouldn't upgrade to a RAID system without the necessary technical support.

How does a RAID system compare to using individual drives? In theory, they're supposed to be much faster. Yet, in theory, a 56 KBPS (kilobits per second) modem is supposed to give you 56 KBPS, but in reality, you're doing well if you're getting a 50 KBPS connection on the Internet. Since testing my programs on different types of drive systems

would belabor the obvious, I instead suggest using my "fast enough" rule of thumb. If your current system runs "fast enough" with the current system, leave it alone. If it takes six hours to run production and you need it to run in one hour, consider upgrading it. Also consider my "parity" rule. If you're going to invest in an applications server with multiple CPUs and 256+ MB of RAM, you might as well go all of the way, buy a RAID system, and make it a "true" server. Why invest in a racecar, but put in a stock four-cylinder engine in it?

√ **FAQ # 4-** If a large SAS software sortsize is good, is a larger one better?

Answer- I've always been interested in this problem because years ago on a different computer, I increased the sortsize and caused my SAS software program to bomb. According to the "SAS Companion for the Microsoft Windows NT Environment" (page 238), "If your machine has more than 6 megabytes of physical memory and you are sorting large data sets, setting the SORTSIZE= option to a value greater than 2M **may** improve performance." "May" is the operative word. It's not clear what the optimal value is. One should assume that any PC, workstation or server, would have far more than 6 megabytes of physical memory available. Here are the results of my tests:

SORTSIZE	Time
1 MB	5 minutes 26.89 sec.
2 MB	3 minutes 11.37 sec.
4 MB	3 minutes 9.42 sec.
8 MB	3 minutes 7.25 sec.
16 MB	3 minutes 11.57 sec.
80 MB	3 minutes 13.48 sec.
MAX	2 minutes 45.23 sec.

Interpretation of results:

More is not always better. Going from the default of 2MB to 4MB only saved a few seconds and at 16 MB, the time began to increase. However, using sortsize=MAX gave the best time, although not breaking any sorting speed records. However, for huge timely sorts, the time saving may be significant. It's not clear what "MAX" is. According to "Optimizing Systems Performance" (page 6), "Max specifies that all available memory can be used." "MAX" is a vague sortsize, that may vary according to the job. Does SAS software use an algorithm to determine what the optimal "MAX" size? Is it at the expense of the total memory? If one is running a SAS job, one would want to use the maximum available memory for a sort, and when that step is completed, the memory should be available for the next step. If "MAX" is a maximum memory used, why isn't it the default value?

To change the sortsize, one can either edit the config.sas file and change the code, "-sortsize 2m" (here is 2 Mb) to a size they prefer. They can also change the sortsize in the global options, after starting up SAS software on the PC. I've tried changing sortsize with a options

statement and within the PROC SORT statement, according to the SAS manuals, but it wouldn't work.

Here are my rule of thumbs:

- 1) Use at least 2 MB of memory (too little may slow up the sorting).
- 2) More than 2 MB of memory may not make much of a difference.
- 3) Don't assign a larger sortsize than available memory (with the standard PC holding 128 MB and more, this problem is not likely to occur).
- 4) Use sortsize=MAX, unless you want to allocate the memory to other resources.
- 5) To be sure of what is optimal for your major applications, test it with different values.

√ **FAQ # 5-** If a large virtual memory page file size is good, is a larger one better?

Answer- No. A virtual memory page file (also known as a swap file) is a means of using the hard drive as memory, when the physical system memory (i.e., RAM) has been used up. The rule of thumb is: 12 MB greater than the system memory. For instance, if one has 128 MB of RAM and applications require more than that, a virtual memory page file allows the applications to use hard disks as "virtual" memory. The major drawback to this type of "memory" is that it's much slower than real memory. If one constantly uses this "memory" (known as disk thrashing), one should consider adding more RAM to the computer. This virtual memory is only a good temporary

solution for inadequate memory. I ran tests on different virtual memory page file sizes and it didn't make a difference. Large page files won't speed up a job, but it may prevent a program from failing from not having enough memory available.

√ **FAQ # 6-** What is aligned data and what does it have to do with optimizing SAS Software?

Answer- Aligning data is a little known means of optimizing how SAS software handles data. Data aligned in 8-byte boundaries can be moved in 2 clock cycles rather than an unaligned rate as slow as one byte at a time (i.e., eight times as long).

How does one align data?

1. keep numeric data at a full 8 bytes (i.e. use a length statement of 8)
2. keep character data in multiples of 8 bytes (i.e., use a length statement of 8 or multiples of 8)

As of this writing, optimizing by aligning data is theory that I haven't yet tested, but my intuition tells me that it should work.

√ **FAQ # 7-What are some simple things I can do to get better results quickly?**

Answer-

1. **Write better code** (that's why you come to SUGI conferences). Efficient code is fast code and simple code is easily maintainable code. For example, sort only when necessary,

minimize passes in reading data (i.e., don't keep rereading the same data, read it once and then do all of the necessary computations then), and use indexes. Unfortunately, with processors speeds doubling every few years, we can write sloppier code and let the processor make up for our inadequate programming skills.

2. **Hire a programmer.** While nonprogrammers can prepare simple SAS applications, it's obvious when an amateur tries to write complicated applications. A seasoned programmer will save you much anguish in the future. Once code becomes institutionalized, it becomes harder to fix, especially partially forgotten historic data structures. If one needs to use several years of awkwardly setup data, it is easier to write code to accommodate the poor design rather than to recreate the entire database system. I could nail some boards together, build a shelter, and call it house. But it wouldn't be much of one. This answer goes with #1- **Write better code**- hire a professional code writer.
3. **Be aware** that hardware and software manufacturers may take advantage of your fast components and piggyback onto them. For instance, modem makers have been leaving some components off their boards, relying instead on your CPU to perform some of the work it formerly did. "Bargain" PCs, in particular, are doing this type of "sharing." Some 32 MB RAM

PCs may really only have 28 MB RAM available for the CPU, since 4 MB of that RAM may be used as graphics memory, rather than the purposefully faster video card SGRAM (synchronous graphics RAM).

4. **Use the right tool for the job.** Don't use a 386 box to run Windows NT worthy programs. Of course, upgrading such a system may be futile, since it may cost more than just buying a new box.
5. **Defragment the hard drives-** Defragmenting a hard drive means having the bytes that make up a file stored contiguously to each other. If the bytes are stored in several places on a hard drive, then there is a "checkerboard" effect, where the computer has to work harder to find the scattered components of a fragmented file. Windows NT 4.0 does not come with defragmentation software but there is third party software available.
6. **Run the various performance counters** available under Windows NT to evaluate and identify your bottlenecks and then upgrade those components.
7. **Learn about optimizing cache-** Originally I was going to do a FAQ on caches, but it would be too complicated for this paper. It deserves a paper in itself. There are many different types of caches to consider, hardware and software, as well as considerations of the different clock speeds of cache on different types of processors. A cache is method of quickly

reusing information without having to reread the same code or data from a hard drive.

In conclusion/final observations

Optimization is a process that never stops- the technology changes too fast. Optimizing SAS on Windows NT is like optimizing almost any application server. More memory and more sophisticated mass storage could produce major improvements in your number crunching. Some factors, such as multiple processors and tweaking options like sortsize may not make much of a difference. Some concepts like 56 KBPS modems work in the laboratory but not in the real world. Test ideas in your own "labs" to see if they make a difference for your applications. My tests apply mainly for my applications but yours may be more CPU or I/O intensive and you'll need to concentrate on those factors. For instance, if you have a lot of data to read and write, upgrading and tweaking your data storage will be a higher priority than improving your processors.

Finally, one should optimize systems toward reasonably practical. Cutting out seconds may make a difference in a road race, but in the real world of data processing, there becomes a point when a reasonable completion time is the only thing that matters.

References:

Andrews, Jean, (1998) A+ Exam Prep, Scottsdale, Arizona: Coriolis Company.

Bardes, David (Dec. 15, 1998) "A Different Kind of Modem", PC Magazine, p214.

Poor, Alfred. (Sept. 1998) "Alfred Poor's Computer Cures." Computer Shopper, p470.

SAS Institute Inc., SAS Companion for the Microsoft Windows NT Environment, Version 6, First Edition, Cary, NC: SAS Institute Inc., 1993. 382 pp.

SAS Institute Inc., SAS web page- Optimizing Systems Performance, Cary, NC: SAS Institute Inc., 1999. <http://www.sas.com/partners/enterprise/ibm/optimize.html>.

Strebe, Matthew, et al, (1997) MCSE: NT Server 4 Study Guide. San Francisco: Sybex Network Press.

Walker, Jamie, (1997) SAS web page- Tuning Your NT Server for the SAS System, Cary, NC: SAS Institute Inc. <http://www.sas.com/partners/enterprise/msinfo/tuning.html>

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

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

Appendix I :Specifications

The Computer used in these tests-

- a. Quad CPU 200 MHz/1024 Kb cache Pentium II
- b. 36 GB RAID Level 5 hard drives (three 18Gb Ultra SCSI hard drives act as one hard drive, and if one drive goes down, a new one can be swapped in its place and the data regenerated onto it from the other two disks.)
- c. 256MB RAM
- d. Windows NT 4.0
- e. SAS 6.12

Test Program Details:

1. 1.2 GB data were read in from twelve permanent SAS datasets.
2. 134 MB of data were parsed out with an "IF" statement.
3. Index files were not used.
4. 134 MB of data was sorted on 2 keys.

Appendix II- Terms Used

CPU- Central Processing Unit- also called microprocessor. This is the computer chip that processes data from memory and the hard drives, crunches numbers and executes software instructions.

Defragmentation- Software algorithm to make files spread out

over a disk as contiguous as possible, to speed up reading these files.

Fault Tolerance- How much a system can "tolerate" system failures such as hard drives crashing. Redundancy (such as multiple drives in a RAID level 5 setup) is one way of achieving adequate fault tolerance.

Hot Swappable- a method of removing a defective hard drive and replacing it with a new hard drive and recreating the data onto it, all without having to shut down the server.

Processor- Another name for CPU

RAID- Redundant Array of Independent Disks- Method of setting up multiple disks to store data as one volume, to improve performance and/or create fault tolerance. There are several levels, based on one's processing optimization needs. Only levels 0 and 5 were mentioned in this paper.

RAM- Random Access Memory. Temporary storage used by computer software on memory chips or modules.

SORTSIZE- option in SAS software to vary how much memory is allocated to a "PROC SORT" step.

Virtual Memory Page File- a software method of letting a hard drive store information that normally would go into RAM. This file is useful when the RAM is full.