SAS SOFTWARE PERFORMANCE
IN THE PERSONAL COMPUTER ENVIRONMENT

Donald R. Deese, Computer Management Sciences, Inc.
John D. Ebner, Litton Computer Services

ABSTRACT

The SAS Institute has ported a number of SAS software products to the personal computer environment. The SAS System Version 6 for Personal Computers potentially offers an inexpensive and convenient interactive environment at the workstation level. However, the potential advantages of the SAS System for Personal Computers can be achieved only if you have a proper microcomputer configuration, use proper supporting software, and select proper SAS configuration options. This paper describes benchmark results illustrating the performance improvements that can be obtained using proper configurations, supporting software, and SAS configuration options.

INTRODUCTION

We are computer performance and capacity planning analysts, often collaborating to analyze the performance and capacity of large IBM mainframe systems. We use personal computers extensively in our projects: performing extensive interactive analysis, developing reports and graphs, writing SAS code to be uploaded to the mainframes for execution, incorporating SAS output into word processors, using SAS/GRAPH to prepare presentations, etc.

When we began using SAS Version 6 for Personal Computers, we installed the product on an IBM PC/XT. We quickly appreciated that an IBM PC/XT is not a platform on which to run SAS. Response to simple SAS PROCs was many seconds or even minutes. We obviously needed better performance.

We decided that there were several ways to gain optimal performance for SAS running on our personal computers: (1) have fast microprocessors, (2) have fast disks, (3) tune the SAS parameters affecting performance, and (4) use disk caching software.

The first two ways to get better performance were accomplished by a "money" solution.

* We purchased state-of-the-art hardware (at the time, the 386 20MHz hardware was the fastest available).

* For our two "main" systems (the PS/2 Model 70 and the ALR FlexiCache 20386) we purchased large capacity and fast (ESDI technology) disks. We also purchased an ALR 386/220 to use as a backup for use on a critical project (we could not afford to lose time on this project because of periodic problems that we had with the ALR FlexiCache 20386).

Although we had acquired the fastest hardware then available, we knew that these systems should be tuned to improve performance. We decided to conduct some experiments with the software. The purpose of our experiments was to determine the optimal setting of the SAS parameters affecting performance. Additionally, we had installed disk caching software to improve performance. However, we needed to know how to split our extended memory between SAS and disk caching.
ENVIRONMENT

We used three 386 20MHz hardware configurations for the majority of our experiments: (1) an IBM PS/2 Model 70 with math co-processor, (2) an Advanced Logic Research, Inc. (ALR) FlexCache 20386 with 32K of 35 nanoseconds caching RAM, and (3) an ALR 386-220. Exhibit 1 illustrates major characteristics of the systems that we purchased.

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>MEMORY</th>
<th>CACHING RAM</th>
<th>SIZE</th>
<th>AVG SEEK</th>
<th>COST</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM PS/2 (M 70)</td>
<td>4 MB</td>
<td>N/A</td>
<td>110 MB</td>
<td>17 MS</td>
<td>$7,136</td>
</tr>
<tr>
<td>ALR FlexCache</td>
<td>2 MB</td>
<td>32K/35 NS</td>
<td>150 MB</td>
<td>18 MS</td>
<td>$7,729</td>
</tr>
<tr>
<td>ALR 386-220</td>
<td>2 MB</td>
<td>N/A</td>
<td>80 MB</td>
<td>28 MS</td>
<td>$3,814</td>
</tr>
</tbody>
</table>

HARDWARE CONFIGURATIONS

EXHIBIT 1

For our basic experiments, we used DOS (Version 3.31) and SmartDrive (Version 2.10) from Microsoft, and QEMM/386 (Version 4.1) from Quarterdeck Software. DOS is a standard operating system for personal computers. SmartDrive is a software driver that caches disk I/O, using expanded memory. QEMM/386 is a software driver for 80386-based personal computers that transforms the extended memory into expanded memory.

BENCHMARKS

We conducted the experiments using three different benchmarks: (1) the TESTBASE code released by the SAS Institute with SAS Version 6 for Personal Computers, (2) the TOWEY benchmark, and (3) a modified TOWEY benchmark.

TESTBASE Benchmark

TESTBASE is a variety of SAS PROCs and SAS code that exercise the Base SAS Version 6 for Personal Computers. Exhibit 2 illustrates the SAS PROCs exercised by the TESTBASE code. We typically use most of the functions tested by the TESTBASE code. Consequently, we felt that the TESTBASE benchmark would fairly represent the type of interactive processing that we generally would perform on our personal computers.

APPEND  CALEND A  CATALOG
CHART   CIMPORT  COMPARE
CONTENTS COPY   CORR
CPORT   DATASETS  DBF
DIF     FORMAT   FORMS
FREQ    MEANS    PLOT
PRINT   RANK     SORT
STANDARD SUMMARY  TABULATE
TIMEPLOT  TRANSPOSE

SAS PROCS EXERCISED BY TESTBASE CODE

EXHIBIT 2

One drawback of the TESTBASE benchmark is that only small files are processed (the TESTBASE code was designed to test function, rather than performance). However, since we use our PCs for interactive purposes, these tests with small files reflect the type of interactivity that we would expect from SAS on our PCs. Further, interactivity was our primary concern; we weren't particularly concerned with tests that ran for 15-20 minutes.

However, we decided to select another benchmark to test larger files. There were two reasons for running the additional benchmark: (1) we felt that the results might be of interest to the SAS user community and (2) we wanted to compare our results with the results reported by TOWEY at SUGI 14.

TOWEY Benchmark

TOWEY executed his benchmark on a variety of personal computers and described the results at SUGI 14. The TOWEY benchmark creates two data files: LONG containing 10,000
records of 228 bytes each, and SHORT containing 100 records of 28 bytes each. After creating the files, the TOWEY benchmark sorts the data. Exhibit 3 illustrates the code contained in the TOWEY benchmark.

```sas
DATA LONG;
  LENGTH CHAR1 $200;
  DO I = 1 TO 10000;
    VAR1 = UNIFORM(938);
    VAR2 = UNIFORM(0);
    CHAR1 = 'ABC....................';
    OUTPUT;
  END;
RUN;

PROC SORT DATA=LONG;
  BY VAR1 VAR2;
RUN;

DATA SHORT;
  DO I = 1 TO 100;
    VAR1 = UNIFORM(938);
    VAR2 = UNIFORM(0);
    OUTPUT;
  END;
RUN;

PROC SORT DATA = SHORT;
  BY VAR1 VAR2;
RUN;
```

The TOWEY LONG benchmark creates fairly large records (one variable in the record is 200 bytes long). Many files processed under SAS on personal computers do not contain such large records. Consequently, we decided to modify the TOWEY benchmark to create a third benchmark. Our modification was trivial: we simply changed the 200-byte variable used by TOWEY's LONG file to be a 20-byte variable.

**Benchmark Timing**

We encountered a problem with timing the benchmarks. In our initial approach, we saved the LOG window resulting from each benchmark to a file. We wrote SAS code to read the file and to extract (1) the PROC or DATA step that executed and (2) the step execution time as reported by SAS. Our code then summed the times and reported the total benchmark execution time.

We quickly discovered a major problem with this approach: SAS records and reports times in seconds, rounded to the nearest second. Some PROC steps would complete in 0 seconds! The times for other steps would vary from one execution to the next, by up to two seconds per execution. We found that the total time for the TESTBASE benchmark (composed of many steps) would vary considerably from one execution to another — based solely upon the imprecise manner in which SAS reported the individual step execution times.

We devised another approach, as shown in Exhibit 4. This approach captured the time at the beginning of the benchmark and at the end of the benchmark. Consequently, the reported benchmark execution time for repeated executions of the same benchmark could vary by a maximum of two seconds from one execution of the benchmark to the next.

```sas
DATA NULL;
  CALL SYMPUT('START',TIME(»);
RUN;

************************
BENCHMARK CODE
************************

DATA NULL;
  FORMAT START STOP ELAPSEO TIME8. ;
  STOP = TIME();
  START = &START;
  ELAPSEO = STOP - START;
  PUT START= STOP= ELAPSEO= ;
RUN;
```

**SOFTWARE TIMING CODE**

EXHIBIT 4

**EXPERIMENTS**

Our experiments were divided into three categories: (1) the basic experiments that tested major parameters affecting the performance of SAS, (2) experiments to test the TOWEY benchmark and modified TOWEY benchmark, and (3) additional experiments to test miscellaneous CONFIG.SAS parameters. This paper
is mainly concerned with the basic experiments that test the parameters affecting the performance of SAS. We performed the additional experiments mostly out of curiosity. Since some of the results are surprising, we decided to include them in this paper.

**BASIC EXPERIMENTS**

The basic experiments measured the effect of altering the "-EMS" parameter in CONFIG.SAS, (2) measured the effect of disk caching, and (3) determined an optimal way to split extended memory between SAS and disk cache.

**-EMS Parameter in CONFIG.SAS**

SAS allows you to use an expanded memory specification (EMS) board by specifying a configuration option (the "-EMS" option specified in CONFIG.SAS). Specifying the -EMS option increases your machine's capacity to execute memory-intensive SAS procedures and increases your capacity to process SAS data sets with large numbers of variables. The SAS Institute states that you can specify "-EMS n" where "n" is the number of 16K pages of EMS allotted to the SAS system. Alternatively, you can specify "-EMS ALL" where "ALL" tells SAS to use all available EMS. (Reference 2 indicates that SAS will use up to 2 megabytes of EMS.)

When you specify the -EMS option, SAS loads several modules into the allotted expanded memory. These modules have been specially optimized for EMS. Thus, three benefits are achieved by using the "-EMS" parameter: (1) optimized SAS modules are used, (2) these modules are resident in EMS and thus do not have to be repeatedly loaded from disk, and (3) SAS can use allotted expanded memory for data storage (e.g., it can use the EMS memory for work space when executing PROC SORT, PROC FREQ, etc.).

Our basic EMS experiments varied the amount of the -EMS parameter. We used EMS values ranging from "0" (indicating no EMS was to be used) up to "128" (the maximum EMS that SAS was said to use: 128 * 16K bytes is 2 megabytes). Note that the "128" value was specified only on the IBM PS/2 Model 70, since the two ALR machines did not have this much memory.

Exhibit 5 and Exhibit 6 show the absolute and performance ratio comparisons of altering the SAS "-EMS" parameter in the CONFIG.SAS file.
These exhibit show that the biggest benefit is obtained by adding 18 pages (288K) of EMS to SAS. (We selected 18 pages because this is the minimum memory that SAS requires in order to load all of its optimized modules. Reference 2 recommends that only 4 pages of EMS be given to SAS as an initial value. However, we could find no improvement with any value less than 18 pages of extended memory.)

Further, adding over 40 pages (640K) appears to have no performance benefits. In fact, we discovered that memory management overhead can adversely affect performance if over 35 pages of EMS are added to SAS.

### Disk Caching

The SmartDrive disk caching software is invoked by specifying "DEVICE=SMARTDRV.SYS" in the CONFIG.SYS file. A parameter allows you to specify how much memory is allocated to SmartDrive. SmartDrive uses the allotted memory to cache disk I/O, thus eliminating many potential disk accesses. We varied the amount of extended memory between 128K (the minimum amount that SmartDrive will allow) up to the maximum extended memory available (3000K for the IBM PS/2 and 1024K for the two ALR machines).

Note that the QEMM software acquires some of the first megabyte above the 640K line and makes it available as expanded memory. However, the amount of "free" memory in the first one megabyte varied between hardware configurations, depending upon the hardware implementation of ROM, the hardware caching, and so forth. For the sake of consistency, we constrained the amount of memory available to exclude the "free" memory acquired from the first one megabyte.

During these benchmarks, we tested the disk caching without allowing SAS to use extended memory. The purpose was to determine what effect using disk caching software would have on the performance of SAS.

Exhibit 7 and Exhibit 8 show the absolute and performance ratio comparisons of altering the amount of extended memory made available for disk caching. These exhibit show that a substantial benefit is obtained by using extended memory for disk caching.

#### Exhibit 7

**Disk Cache Tests - Varying Cache Size**

<table>
<thead>
<tr>
<th>Cache Size</th>
<th>Performance Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>2</td>
<td>0.8</td>
</tr>
<tr>
<td>3</td>
<td>0.7</td>
</tr>
<tr>
<td>4</td>
<td>0.6</td>
</tr>
<tr>
<td>5</td>
<td>0.5</td>
</tr>
<tr>
<td>6</td>
<td>0.4</td>
</tr>
</tbody>
</table>

---

#### Exhibit 8

**Disk Cache Tests - Varying Cache Size**

<table>
<thead>
<tr>
<th>Cache Size</th>
<th>Performance Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>2</td>
<td>0.8</td>
</tr>
<tr>
<td>3</td>
<td>0.7</td>
</tr>
<tr>
<td>4</td>
<td>0.6</td>
</tr>
<tr>
<td>5</td>
<td>0.5</td>
</tr>
<tr>
<td>6</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Since the two ALR machines have only two megabytes of memory, only the results of using 1024K for disk caching are shown in Exhibit 7 and Exhibit 8. We continued the experiments with the IBM PS/2 Model
The performance of the IBM PS/2 Model 70 continued to improve until about 1500K of extended memory was made available for disk caching. After this point, very little improved performance was obtained.

Varying EMS and Disk Cache

We have a fixed amount of extended memory available on our machines (3 megabytes on the IBM PS/2 and 1 megabyte on the two ALR systems). We needed to determine how to split this memory between SAS -EMS and disk caching. We had discovered that the most dramatic improvement came with specifying "-EMS 18" (allocating 288K of extended memory to SAS), and that no improvement was obtained unless at least 288K was made available. Thus, we conducted experiments varying the allocation of extended memory between "-EMS 18" (or 288K) and "-EMS 56" (or 896K) for SAS. At the same time, we varied the allocation of extended memory between 736K and 128K for disk caching.

Exhibit 9 and Exhibit 10 show the absolute and performance ratio comparisons of altering the amount of extended memory made available for disk caching and providing the remaining extended memory to SAS.

These exhibits show that the biggest initial benefit is obtained by adding 128K for disk caching. With this amount of extended memory made available for disk caching, there were 50 pages (or 800K) available for SAS. However, we had earlier determined that 18 pages (or 288K) of extended memory was the minimum that SAS required for good performance, and that SAS could not use over 40 pages (or 640K) of extended memory.

Performance continued to improve until we allocated 736K of extended memory to disk caching and allocated the remaining 288K (18 pages) of extended memory to SAS. We conclude that with 1024K of extended memory available, the optimal allocation is 288K allocated to SAS ("-EMS" set to 18 in CONFIG.SAS) and the remainder allocated to disk caching.

TOWEY Benchmark Results

At SUGI 14, TOWEY presented results from benchmarking a variety of personal computers, using the benchmark shown in Exhibit 3. The most controversial part of his paper was that he showed an 8MHz PC/AT with EMS as the fastest configuration in executing his benchmark. Additionally, other results were confusing (e.g., he reported that...
the benchmark execution times varied by almost 100% depending upon which logical disk drive was used for SASWORK files. We executed the TOWEY benchmark and the modified TOWEY benchmark on our systems.

Exhibit 11 shows the results of executing the TOWEY benchmark and Exhibit 12 shows the results of executing the modified TOWEY benchmark, with various values of the SAS "-EMS" CONFIG.SAS parameter and varying amounts of memory allotted for disk caching. The times are reported in minutes.

<table>
<thead>
<tr>
<th>EMS</th>
<th>CACHE</th>
<th>ALR-386</th>
<th>ALR-386</th>
<th>IBM PS/2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>15.03</td>
<td>15.40</td>
<td>15.15</td>
</tr>
<tr>
<td>18</td>
<td>736</td>
<td>11.21</td>
<td>9.95</td>
<td>10.36</td>
</tr>
<tr>
<td>30</td>
<td>544</td>
<td>10.33</td>
<td>10.05</td>
<td>10.41</td>
</tr>
<tr>
<td>40</td>
<td>384</td>
<td>10.32</td>
<td>9.05</td>
<td>9.69</td>
</tr>
</tbody>
</table>

**TOWEY LONG SORT BENCHMARK RESULTS**

**EXHIBIT 11**

<table>
<thead>
<tr>
<th>EMS</th>
<th>CACHE</th>
<th>ALR-386</th>
<th>ALR-386</th>
<th>IBM PS/2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>3.57</td>
<td>2.32</td>
<td>2.65</td>
</tr>
<tr>
<td>18</td>
<td>736</td>
<td>2.58</td>
<td>2.33</td>
<td>2.27</td>
</tr>
<tr>
<td>30</td>
<td>544</td>
<td>2.78</td>
<td>2.14</td>
<td>2.26</td>
</tr>
<tr>
<td>40</td>
<td>384</td>
<td>3.31</td>
<td>2.61</td>
<td>2.62</td>
</tr>
</tbody>
</table>

**MODIFIED TOWEY LONG SORT BENCHMARK RESULTS**

**EXHIBIT 12**

These exhibits are shown in tabular form since this is how TOWEY displayed his results. Additionally, only summary results are shown, rather than showing the DATA and SORT step separately.

Exhibit 11 shows that the results from our running the benchmark almost exactly match those reported by TOWEY, for the IBM PS/2 Model 70 without EMS or disk caching memory. TOWEY did not report results for ALR machines, so the values shown in Exhibit 11 are not comparable. However, the benchmark results shown in Exhibit 5 to Exhibit 10 show that the ALR FlexCache is roughly comparable to the IBM PS/2 Model 70. So it is plausible that the poor performance reported by TOWEY is to be expected when running a 386 machine without EMS and without disk caching. However, providing EMS and disk caching result in significant improvements.

After some reflection, it is clear that TOWEY's LONG benchmarks mostly test rudimentary I/O capability of the configurations, rather than testing the capability of the processors.

* The benchmark consists of a DATA step and a PROC SORT step.
* The DATA step certainly tests only I/O, since it simply creates a 10,000-record file.
* With limited memory available in which to sort, the PROC SORT step clearly doesn't tax the processing power of the 386 machines. Sorts generally are I/O bound operations, and sorts with limited memory certainly are I/O bound operations.

Since the fastest configuration reported by TOWEY (the 8MHz PC/AT) included 384K of EMS, this configuration provided more memory for sorting and possibly disk caching. Consequently, less disk I/O was required. We conclude that the results reported by TOWEY basically compared the memory speed of the 8MHz PC/AT with the disk speed of the 386 machines. These results do not properly reflect the speed of 386 machines in a real world environment.

Note that the performance shown in Exhibit 11 improved slightly as the "-EMS" parameter in CONFIG.SAS was increased (up to the 40-page maximum). However, the performance
shown in Exhibit 12 deteriorated as the "-EMS" parameter in CONFIG.SAS was increased. We attribute this to interaction between the memory management overhead associated with extended memory versus the overhead associated with disk caching.

The results shown in Exhibit 11 and Exhibit 12 reflect the total time for TOWEY's LONG and SHORT benchmarks. We noticed that performance of the SHORT benchmark improved far more dramatically than these total times reflect. Since we are primarily concerned with interactivity, the SHORT benchmark would tend to better reflect our typical file size.

Exhibit 13 shows the benchmark times for just the SHORT benchmark. We have already shown that "-EMS" set to 18 and the rest of extended memory used for disk caching is the best for our systems. Consequently, Exhibit 13 simply reflects the percent improvement between the "worst" and the "best" parameter specifications. The performance improvement varies from 2.75:1 for the ALR FlexCache to 4.60:1 for the ALR 386/220! (Recall that the ALR FlexCache has 32K of 35 nanosecond RAM for hardware disk caching. Thus, its "worst" times were not as bad, and its percent improvement was not as great as the improvement for the other 386 machines. Still, the 2.75:1 performance improvement for the FlexCache is quite dramatic!)

We were never able to duplicate TOWEY results of widely varying performance based upon which logical disk drive contained SASWORK. TOWEY reported results varying by over 10 minutes depending upon the logical drive containing SASWORK. We executed his benchmark on our systems in different logical disk partitions (we varied the logical partitions between C Drive, D Drive, and E Drive). None of our results varied more than a few seconds within a particular machine as a function of the logical disk drive, even with a zero "-EMS" value in CONFIG.SAS and disk caching turned off. We conclude that perhaps TOWEY encountered experimental error in this aspect of his work.

ADDITIONAL EXPERIMENTS

As mentioned earlier, we performed some additional experiments, testing various CONFIG.SAS parameters. For the sake of brevity, only summary results are discussed.

-FILECACHE Parameter

We tested various options of the "-FILECACHE" parameter in CONFIG.SAS, with our disk caching software allotted 736K of extended memory and the "-EMS" parameter set to 18 pages. These experiments showed that there is no measurable performance effect in using the -FILECACHE parameter so long as disk caching software is operational.

-FILEBUFFERS Parameter

We tested various options of the "-FILEBUFFERS" parameter in CONFIG.SAS, with our disk caching software allotted 736K of extended memory and the "-EMS" parameter set to 18 pages. These experiments showed that there is no measurable performance effect in specifying file buffers to SAS, so long as disk caching software is operational (unless extreme values are specified, in which case performance severely deteriorates).
This is to be expected, since the disk caching software basically performs the same function that the SAS "FILEBUFFERS" performs.

Use of Extended Memory by SAS

Reference 2 (page 202) states that SAS will use up to two megabytes of extended memory. We tested the use of extended memory by SAS. We found that SAS will accept a specification of up to two megabytes of extended memory, and will "use" the memory in the sense that the memory is unavailable for use by any other application. However, SAS apparently does not actually use more than 640K of the memory for performing work.

We tested this hypothesis in several ways:

* We executed the TESTBASE benchmark, specifying 50, 60, 70, 80, 90, 100, and 110 pages as the values for the "EMS" parameter (this test was executed on the 4 megabyte IBM PS/2 Model 70). There was no performance improvement beyond 40 pages (or 640K of memory).

* We used PROC FREQ with TABLES X*Y*Z for a large file. With "EMS" set to 30 pages, SAS began removing variables from the distribution at a certain level. SAS began removing variables at the same level for "EMS" set to 40 pages, 50 pages, etc. We concluded that SAS will use only 480K of extended memory for PROC FREQ.

* Our tests of PROC SORT of a large file showed that there is no performance improvement when more than 40 pages have been specified for the "EMS" parameter. We concluded that SAS will use only 640K of extended memory for PROC SORT.

* We ran the Test Library issued with SAS/GRAPH for Personal Computers. We could find no SAS PROC that would use more than 30 pages (480K of memory).

As a result of these test, we conclude that SAS will use between 480K and 640K of extended memory, but will not actually use more than 640K of extended memory to perform work.

Subsequent conversations with the SAS Institute confirms this conclusion. However, we were told that SAS can use additional memory to store PROC code, saving time required to reload the code. Using memory in this fashion did not yield a measurable performance improvement in our benchmark tests.

Recommendations

We strongly recommend that a megabyte of extended memory be added to any personal computer executing SAS. A significant performance improvement is obtained by allocating this megabyte of extended memory to SAS and disk caching. Our results show that almost 5:1 performance ratio can be gained by properly using a megabyte of extended memory.

Our experiments show that the extended memory is best used by allocating between 288K and 480K to SAS. This is accomplished by specifying a range of "EMS 18" to "EMS 30" in CONFIG.SAS. The amount that should be specified in this range depends upon your individual applications. All remaining extended memory should be allocated for disk caching, unless you require that some of the memory be available for other software (e.g., a word processor or spreadsheet application).

Our results show that there is no significant advantage in having more than one megabyte of extended memory, unless you require the memory for some other application. SAS will not use the additional
memory, and only a modest improvement is obtained using an additional megabyte of extended memory for disk caching. In fact, there was no performance improvement gained from having more than two megabytes of extended memory.

The results show that, for the benchmarks used, the ALR 386/220 system had a much better cost/performance than did our two "main" computer systems. In fact, with optimal parameter settings, its absolute performance was almost as good as the systems costing twice as much! We were chagrined at these results, since the ALR 386/220 was purchased only as a backup system. We rarely use the ALR 386/220 for actual project work.

Upon reflection, perhaps we should have anticipated this result: most of our work is disk-intensive with relatively small files, the Intel 80386 chip contains a very powerful CPU, and disk caching allows data to be available when needed. Thus, the expensive items in our "main" computer systems (the ESDI large capacity disks and the hardware caching) added several thousand dollars to the cost of these systems. Neither of these expensive items seem to be worth their cost for our SAS applications.

SUMMARY

When we began our experiments, we had no idea that we would encounter such tremendous performance improvements simply by altering two parameters: the "EMS" parameter in CONFIG.SAS and the amount of memory allotted for disk caching. However, this initial impression was changed as we conducted the experiments. However, we could actually observe the improvements as the benchmarks executed: SAS PROCs that earlier required 3 or 4 seconds to execute suddenly executed in 1 second or less! Such dramatic differences in performance are extremely important in any environment. In an interactive environment, this type of performance improvement is startling.

We are pleased with the results of our benchmarks, and are now extremely satisfied with the performance of SAS Version 6 for Personal Computers. We hope that our results will help increase your satisfaction with using SAS at the workstation level.

References
1 TOWEY, C. Brian, "Benchmarking the SAS System, Release 6.03" SUGI 14 Proceedings, 1989

Authors Contact
Donald R. Deese
Computer Management Sciences, Inc.
6076-D Franconia Road
Alexandria, VA 22310
(703) 922-7027

John D. Ebner
Litton Computer Services
1831 Michael Faraday Drive
Reston, VA 22090
(703) 471-9200