PERFORMANCE WITH SAS

Mogens Weinreich
The Danish National Institute of Social Research

In several institutions, goverment and private with extensive use of computer processing the organisation now tends to be structured in two subdivisions dealing with systems and information to users respectively:

1. The Systems Section

This section is responsible for the hardware and the software being used. The staff concentrates its efforts on keeping things going as smoothly as possible. Some computer staffs try to tune the computer system by different performance tools, like the Merrill code on TSO or SAS/VMAP on VM/CMS, while others just buy more and more hardware without knowing that the old installation could have been sufficient if properly used.

2. The Information Center

This section is the center for the end-user support. The staff helps to construct easy-to-use programmes, gives courses etc. The section is also responsible for examining new software packages. The section is very often cooperating with the users in a gentle atmosphere.

This organisation which comprises a Systems Section and a user orientated Information Center/User Service Center has a weakness, however, that I would like to point out: The stock of hardware tends to expand very fast, which is of course very expensive. This fact is a result of the dual organisation mentioned above. The staff of the Information Center uses all of its time selling solutions and end-user tools which are heavy consumers of computer time, rather than making programmes perform well on the computer system as performance is thought to be the responsibility of the Systems Section.

The users

The user (the end-user) is a person outside the technical organisation described here. The group of users is quite a heterogenous one. Some users have a good background in computer work and need very little support from the Information Center. Other users badly need that support, and this group is very often the biggest one. The user group often claims for 50 per cent or more of the available computer resources. If you have to wait too long for the computer to respond, you will obviously go to the Information Center complaining: Why do I have to wait for my results that long? etc. The Information Center staff is of course very unhappy about this, as they are trying to service their customers (the users) efficiently. The result is often that the head of the Information Center immediately complains officially to the person responsible for the technical staff in the Systems Section. And now the whole organisation approaches the Managing Director with this urgent and serious situation: They can in every respect document that the available computer system is insufficient for the tasks needed. Nobody outside this organisation is able to analyse these claims and a heavy pressure is put upon the Managing Director. The decision can very well be to order a bigger and more powerful system and install it

as fast as possible. The new system is often five times or more bigger than the old one. It has been costly, but for a time peace is restored, everyone is satisfied as the response time is satisfactory, at least for a while.

The trouble is that nobody realized that the real problem was a wrong policy. If instead, the right policy had been conducted a lot of money could have been saved. Everyone tried to do his best, but it did not work out due to insufficient communication between the Information Center and the Systems Section.

The real problem is that about 90 per cent of the programmes executed on the computer are garbage that never should have been started! The Information Center should be selling well performing programmes, and not just solutions to end-users problems.

An Example

A menudriven system on VM/CMS with ISPF screens is easily created. Very often, however, it kills the computer. The same application without menues is much quicker. A test on our IBM/4331 4 Megabytes shows that we could have 5 users at a time with an application driven under ISPF-menues; but 25 users on the same application without ISPF!

The benefit from a menudriven system is obvious, but the cost-benefit ratio should be analysed for such a system; and in the technical organisation above there is none to take a decision to that end. The Information Center is responsible for this development; the center is mainly thinking of having satisfied customers, and removing menues etc. will be of no benefit to this end. This policy is indeed a very expensive one for the Institution. Of course the computer manufacturers are interested in exactly this development; and a clear symptom of this is that the whole idea of establishing an Information Center is conceived by a computer manufacturer knowing that an organisation selling programme solutions will tend to use the computer system more extensively regardless of costs. (Organisation structure see appendix, figure 1.)

Our organisation

In The Danish National Institute of Social Research we have the very important link between users, Information Center and technical staff, and the following section will show why this is so important.

In January 1983 we installed an IBM 4331/4M with 3 3370's disc stations. However, we only have one programme installed to analyse data, namely the SAS-package: SAS/BASIC and SAS/FSP.

In the organisation the technical staff is a natural part of the Information Center. Due to the small scale computer system we often have to make performance considerations to provide all of our users with a good service.

An easy way to do this is of course to upgrade the computer to the new 4361 which is five times faster. Another way is to concentrate efforts on internal education of the users and on tuning the performance of the software as well as the hardware.

Performance Considerations

Our problem is simply to get SAS running as fast as possible.

Improvements

To increase the efficiency of the SAS system the SAS Institute recommends to install the SAS data-supervisor and the procedures in DCSS (DisContinuous Saved Segments). This is an easy task to carry out due to the very good documentation enclosed with the SAS installation tape. The idea of DCSS is twofold: The programme is loaded into storage significantly faster due to the fact that a read I/O-operation is much slower than a page-in operation. Secondly because re-entrant portions of code can be used simultaneously by several users, thus giving a significant reduction of storage consumption. This reduces the overhead costs of paging significantly. To install SAS in segments is important and may mean the difference between a system with a very bad performance and a well performing one.

PL/1 Transient Library

Before SAS can be used, you have to lease the PL/1 transient library from IBM. The library contains a number of subroutines used in the procedures. The library is placed on a minidisc as a txtlib. In our Institute we became aware of an improvement to CP and CMS which made it possible to install this library as a DCSS. It is known by a lot of PL/1 users that doing this can increase the performance of PL/1-programmes by up to 30 times!

This fact and the fact that the SAS-procedures are written in PL/1 persuaded us to try the DCSS solution. But our efforts turned out to be a waste of time. The SAS-people had already done it. Only a few routines were kept on disc so that you were forced to buy the originally code from IBM.

CMS Profile

Log on to the VM/CMS system causes the PROFILE EXEC to be executed. This profile very often enables you to link to hundreds of files, e.g. the CMS-help functions which count for about 1,400 files placed on the same disc. This can reduce your virtual storage by hundreds of kbytes. When access to a disc is established VM creates an internal directory in your addressable storage. This decreases the addressable space by 64 bytes for each accessed file, e.g. the CMS-help functions reduce your virtual storage by nearly 100 kbytes. Instead you should create an EXEC to link and access the specified disc when needed; and after use it should be detached and released. Be sure that the CMSSEG is installed above the virtual storage of your machine. The CMSSEG is a DCSS which improves SAS-performance significantly, as it helps OS-simulation.

SAS Profile

In the SAS profile the options of the SAS system are available to the user. The profile is very often designed by the technical staff and it is frequently set to the default

values, despite the fact that a careful decision can improve overall performance. If, for example, your job only uses the procedures from SAS/BASICS, you will achieve more addressable space, if you refrain from loading the FSP and GRAPHICS-modules. In addition it makes performance slightly better. If you are not using the macro facility either, you can avoid loading it by the option NOMACRO.

Our standard profile contains:

```
NOFS
NOFSP
NOMACRO
```

and our users know that they have to add the relevant facilities themselves, if they want to use them, like:

```
SAS SASJOB (FS FSP
```

This extended SAS profile improves overall performance by approx. 5 per cent and decreases the need for storage with 50 kbytes.

If you use a temporary disc as SIODISC, it should be formated with the highest possible SIZE, e.g. 4,000 bytes. If your disc is big enough, you should consider the SAS option BLKSIZE=32000, which also improves performance slightly, especially on big data sets.

As we are using the VM/CMS system, I have not mentioned the DOS, TSO and the other versions of SAS. In my opinion the SAS Institute has shown great efforts to get the SAS system to perform well. It is obvious that SAS performs at least ten times faster than an ADI/APL system.

Using SAS

After half a year of using SAS we were all pleased and impressed by how strong a tool SAS turned out to be. However, the computer system was running daily at 100 per cent of capacity causing a total computer degradation. 10 per cent was used by the system just to handle USER-paging etc., even with the SAS-segments installed.

A solution could be to buy a bigger computer or up-grading to the 4361 model. We spent a lot of time tuning the system, but this was not enough. Then we got a piece of good advice from the person in charge of the installation at the Copenhagen Handelsbank. He asked us two questions: "Are you sure that your users are constructing their programmes in an efficient manner?" and "are you sure that all the jobs are necessary?"

The SAS applications guide inspired us to examine the possibilities of reducing the CPU-time, which was our bottleneck.

The fact is that a few lines of SAS code generate an enormous numbers of I/O-operations and an unnessary CPU-load on the system. A careful analysis shows that the Procedure as well as the Data step can be used in a much more efficient way.

The following examples show how much:

The Data Step

The example is a Data set with 3,354 observations and 567 numerical variables, here called US.DATA. We are interested in the 10 first variables.

The first example shows how much the KEEP statement influences the CPU-time.

JOB	CPU - SEC			STORAGE
	1000 OBS	FULL DATA		
		8000 B	32000 B	
1 DATA A; SET US.DATA;	35.4	98.0	93.3	640K
2 DATA A; SET US.DATA (KEEP=V1-V1);	5.6	14.8	14.6	576K
3 DATA A(KEEP=V1-V10); SET US.DATA;	25.2	64.2	62.2	640K
4 DATA A; SET A;	2.6	6.7	6.3	576K

Explanation:

The 8000 B and 32000 B means that BLKSIZE is 8000 bytes and 32000 bytes respectively. A minor improvement on CPU-time is obtained for the big data set, on example 1 about 5 per cent. It seems that the SAS standard choice on 8000 bytes is adequate.

Of course you have to compare these results with the results obtained on your computer, but the pattern should be the same. It turns out to be a clear advantage to pick out the needed data and save them for further analysis in a minor copy. The CPU-time consists of two parts: Namely the I/O-time and the elapsed time used to convert the variables to the floating point. The last job just copies the 10-variable dataset and this is of course the fastest way to do it. The difference between Job 2 and Job 4 could give an estimate for the I/O-time: 14.8 - 6.7 or around 8 seconds, just to handle I/O requests. Please also note the reduction in storage needed to complete the job; this is very important to a storage constrained system.

Using the KEEP and/or DROP options is clearly advantageous. The advantage is of course a function of the number of variables in the data set. However, it very often happens that you have a big data set accessable to several users in read-only mode, while the individual user only needs a few of the variables available in the data set. In this situation it is obvious to use the KEEP option. At our place the rule simply is: "If you need a subset of the variables, use KEEP".

The Procedure Step

We are still using the same data set as the example. The procedures used are PROC FREQ, PRINT and MEANS, which I assume that everybody knows.

JOB	CPU-TIME		STORAGE
	1000 obs	3354 obs	
PROC PRINT DATA=US.DATA(KEEP=V1-V10);	27.1	83.9	640K
PROC PRINT DATA=US.DATA;VAR V1-V10;	32.1	91.3	704K
PROC FREQ DATA=US.DATA(KEEP=V1-V10);	23.1	73.1	670K
PROC FREQ DATA=US.DATA;TABLES V1-V10;	28.2	79.4	670K
PROC MEANS DATA=US.DATA(KEEP=V1-V10);	8.6	26.0	618K
PROC MEANS DATA=US.DATA;VAR V1-V10;	13.6	30.5	640K

This example shows how important it is already at the Procedure step to remember to keep only the variables to be analysed. If the Procedure step contains a VAR statement, as in MEANS and PRINT, the KEEP option is to be preferred to the VAR statement. On a heavy loaded computer the storage consumption is also very important, and here the PRINT and MEANS jobs can be executed succesfully in a slightly smaller region using KEEP compared to using VAR; this reduces paging. PROC FREQ needs the same region in both cases, as the number of distinct categories is the same.

The advantage in time seems to be approximately 5 seconds in both cases.

A SAS Programme

A SAS programme can consist of several Data steps and Procedure steps. At first the user will of course have some syntax and perhaps also logical errors in the SAS code. Then the user tries, maybe several times, to correct the code to get it running correctly, and by doing this he uses a lot of CPU-time.

Syntax Errors

To correct the syntax errors we are just putting

```
OPTION OBS=0;
```

at line 1 in our job set-up. The SAS supervisor will then compile the set-up and you will get some error messages in the SASLOG.

After correcting the syntax errors, next step is:

Logical Errors

This can of course be a more difficult task to handle, but very often the logical error can be detected by executing the job on a few observations. We use to set

```
OPTION OBS=100;
```

in line 1, when testing the SAS code for logical errors. This saves a lot of computer time.

No Errors

When everything is corrected we remove the

```
OPTION OBS=100;
```

and the result often is that the overall load for this SAS programme development has been decreased by a factor 10!

Character Variables Versus Numerical Variables

The SAS applications guide tells the user that character variables are to be preferred in order to increase performance. The reason is that SAS then avoids converting to double precision. If your data are used only to categorize or your variables are forming natural intervals, they can be used as character variables.

The next example shows how important it is carefully to decide whether a variable shall be defined as a numerical or a character one. The variables sex, age, and opinion fall in 2,12, and 2 categories respectively on a data set comprising 2,500 observations.

	JOB	CPU SEC	STORAGE
1	PROC FREQ DATA=NUM.DATA;	8.1	936K
2	PROC FREQ DATA=CHAR.DATA;	4.8	936K
3	PROC FREQ DATA=NUM.DATA; FORMAT V1 SEX. V2 AGE. V3 OPINION;	8.4	936K

Comments:

NUM.DATA contains just the numerical values and CHAR.DATA contains exactly the same data, the only difference being that they are interpreted as character variables. It turns out to be a real advantage to use character variables. The reduction of computer time will often be more than 35 per cent. Job 3 is an interesting one, as the PROC FREQ reads the numerical data, but recodes the variables according to the Format statement. The specified format is read from a format TXTLIB, which is defined by PROC FORMAT. The formats for sex, age, and opinion just convert the numerical 1 to the character '1' etc. The elapsed time turns out to be exactly the same as on the numerical example (JOB 1). The reason is that PROC FREQ runs significantly faster on character variables, and despite the fact that every variabel has to be recoded by the Formats, it ends up with the same elapsed time. For this reason we recommend the use of formats to recode a variable instead of using a DATA; SET;-step.

Multiple Data Set

On some occasions a set-up reads from the same data set several times. By creating a multiple data set a significant benefit can be obtained.

As an example:

A user wants to run PROC FREQ on 5 distinct subgroups of the data. A natural set-up would be:

```
DATA A; SET BACKGROUND;
   IF AGE<65;
PROC FREQ; TABLES SEX*OPINION;

DATA B; SET BACKGROUND;
   IF EARNING>10000;
PROC FREQ; TABLES SEX*OPINION;

etc.
```

We have tried to rewrite this set-up using the multiple data facility.

```
DATA A B C D E; SET BACKGROUND;
IF AGE<65 THEN OUTPUT A;
IF EARNING>100000 THEN OUTPUT B;
PROC FREQ DATA=A;
etc.
```

The Result:

JOB	CPU IN SEC	STORAGE
SINGLE DATA SETS	143.3	960
MULTIPLE DATA SET	24.1	896

It is obviously a great advantage to remember the multiple data facility. In this example it reduces CPU-consumption by a factor 6.

The PL/1-Transient Library as DCSS

As the major part of the PL/1-Transient Library is installed along with the SAS segments, there is only little benefit to be derived from this.

The TESTBASE programme issued on the SAS tape was tried with and without the library as DCSS.

The result was of course quite disappointing; only a few of the procedures showed better results, and then only slightly better. PROC PRINT on small data sets showed the biggest improvements - and this was only a 10 per cent decrease by using DCSS.

If SAS is not installed in segments there is, however, advantages. As an example:

A small job with PROC PRINT.

JOB	SAS-SEGMENTS	PL/1-SEGMENTS	CPUSEC	STORAGE
1	SHARED	SHARED	0.68	576
2		OFF	0.72	576
3	OFF	SHARED	0.61	896
4		OFF	1.40	896

The reason is as follows. Using SAS-segments gives a small overhead, as CP has to build up internal swap-tables to the point where SAS is loaded. PROC PRINT uses some functions from the PL/1-TXTLIB, which is not in the SAS segments. This causes SAS to search for the function on a disc. Job 3 is the fastest due to the fact that there is no in-page overhead and no disc search for the PL/1 routines.

The conclusion is that no benefit is derived from using the PL/1 transient library as DCSS when SAS is running with segments.

SAS Invocations

SAS reports the consumption of CPU-time for every Data and Procedure step. The overhead of invocation of SAS is, however, not reported.

The measurements of this overhead are found by using the CMS commands Q TIME and INDICATE USER.

JOB	TYPE	CPU-SEC	SIO
SAS	SAS IN SEGMENTS	4.9	164
SAS	SAS WITH PSEG AND SSEG OFF	6.5	244

This is another important reason to running SAS in segments. The overhead of invocation is reduced by some 30 per cent, and the number of SIO-requests is reduced by approximately the same.

Conclusion

By an intensive internal education we have succeeded in decreasing the CPU load by at least factor 2! Our IBM 4331 therefore, seems to be adequate for solving our problems for a long time ahead. Our users are still achieving the same services, but now they are not loading the computer with unnecessary jobs. It is my point that this is achieved only due to the narrow cooperation between the technical staff and the Information

Center. The staff in the Information Center shall be aware of their dual responsability: Making well performing programmes as well as helping the users to solve their problems efficiently. The whole organisation benefits from this and a lot of money can be saved, leaving them available for other purposes.

FIGURE 1

Organisation structure

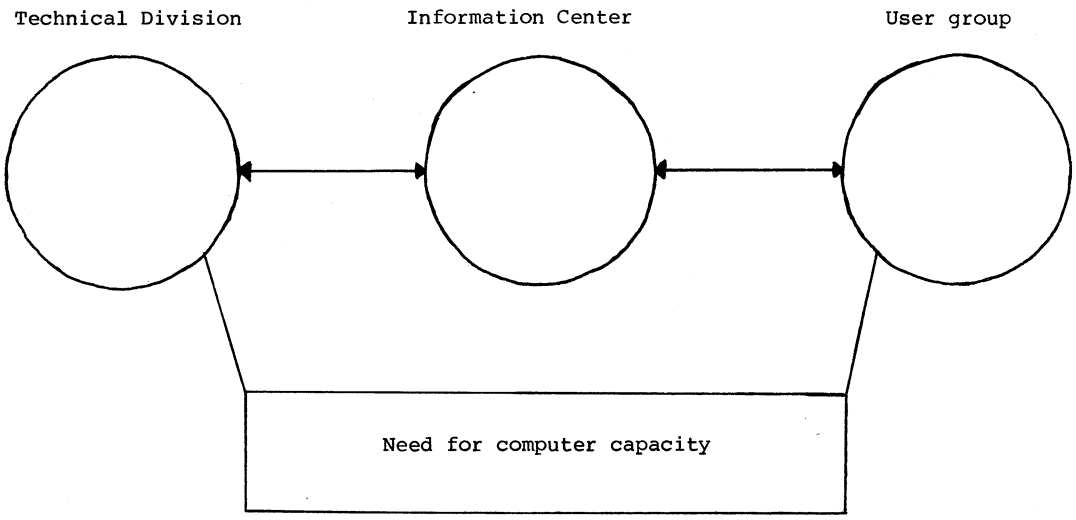
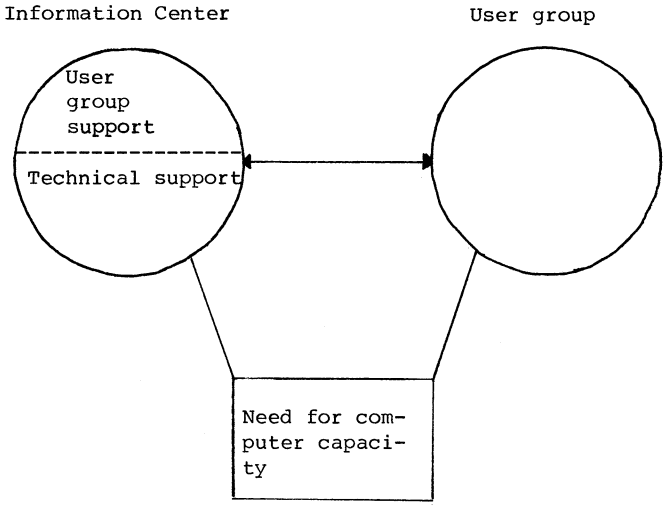


FIGURE 2



An organisation structured as in Figure 1, tends to use a lot more of CPU than does the organisation i Figure 2. The service to the user group will be the same in both situations.