Optimizing Your SAS Performance with Grid Computing

Sy Truong, Meta-Xceed, Inc, Milpitas, CA

ABSTRACT

Google has become very successful by developing an efficient search engine running on commodity hardware. It no longer uses the old model of putting all its resource onto one super computer, but rather it spreads that processing onto a cluster of smaller machines running in parallel to form a grid. Gordon Moore made an observation in 1965 predicting that the number of transistors per square inch used on computers would double every year. This trend has become law and continues to elevate the ubiquitous and relatively inexpensive desktop and laptop computers. This paper will discuss how you can cluster computers in a grid to optimize the execution of SAS programs. Some of the techniques discussed include:

- Implementing supercomputer power with commodity hardware
- Submitting SAS programs sequentially while maintaining inter program dependencies
- Threading multiple groups of programs for optimal performance
- Measuring SAS performance with Statmark™, a standard metric for a cross platform benchmarking for SAS processing
- Scheduling the execution of programs in a grid environment

In the world of Moore's law, it makes less sense to lay out large capital investment for a server. Clustering inexpensive smaller machines and dynamically adding new computers to this architecture within a grid can scale your SAS computing resources to become the Google of search engines.

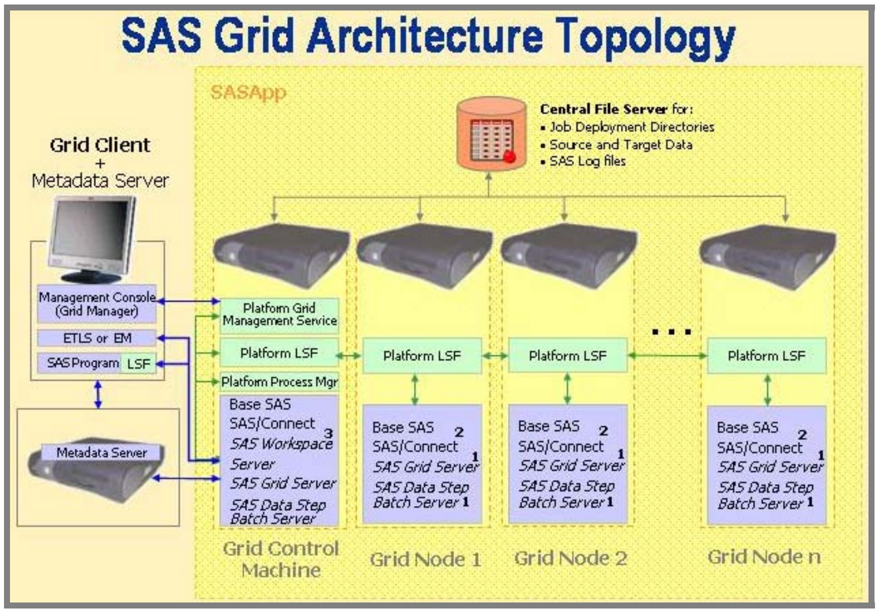
INTRODUCTION

In the space of analytics as statistical models get more sophisticated and the datasets gets larger, computing resources is much needed engine that delivers results. SAS has evolved along with hardware systems to utilize the horse power needed to crunch the statistical models and data manipulations. When I first started working with SAS, it was on a main frame computer system running TSO. This was centrally controlled with very limited user customization from a dumb terminal. As computing chips got smaller, the processing of SAS started to move toward smaller UNIX servers. Then the introduction of SAS on personal computers dramatically changed how most users performed their data exploration. Users were testing out their data models and reports on their PCs, although they still executed things on a networked server for production jobs. This evolution is continuing as the desktop is becoming more powerful. With maturing technologies used to connect these desktop computers, PC desktops are beginning to form computing grids that can outperform the traditional servers. The forces that drive this include the shrinking size and cost of computer chips while performance is increasing. This is coupled with the lowering cost of memory and storage. These combined elements supply analytical tools such as SAS with greater abundance of computing resources. We are at a juncture in this evolutionary stage where the ways the computing resources are utilized can be more important than just obtaining the resources.

IT managers need to evaluate the cost of the lifetime of a server since the price to performance ratio of the computing resources would diminish over time. It is similar to purchasing a car in that the performance of the car does not go any faster but the value of the car is constantly going down. Computing resources have an even lower return on investment in that they become obsolete very quickly as the next model is usually cheaper, yet outperforms the current server model. It is therefore not always prudent to put out large capital expenditures on a piece of hardware when its performance to price ratio will diminish in such short spans of time. Grid computing offers a different model in that commodity hardware can be expensed with less cost. There is greater flexibility in that the grid can scale to match the performance of a growing group without necessarily throwing out the old server for replacement of the new. Nodes can be added and older nodes can be taken off like a living organism shedding dead skin. In the Grid, the newer nodes have the advantage of obtaining the fastest computing power for the cost at that time. This spreading out of the capital expenses on computing resources is analogous to the time valued benefits of spreading out your investments and investing small amounts over your lifetime to form a balanced portfolio instead of putting one big sum investment into a single stock. It acts as a buffer towards the ups and downs of the markets. In this case, it is not the financial market but rather the market of computing hardware cost. As hardware costs continue to get cheaper per price performance, the cost of software seems to get more expensive as the complexity of the software increases. Licensing SAS is not cheap so it is wise to optimize the hardware which SAS runs on since over time, the hardware cost will be a fraction compared to the software cost.

One of the key components in the optimization of computing cost is the ability to measure with precision the performance of your system. This metric can help you evaluate the return on your investment. Without any form of measurement, it is like shopping for a credit card without having the ability to know what the APR or interest rates are. This paper will premiere a free utility called Statmark™ by MXI that will allow you the tools to make the right decision in hardware implementations.

SAS Institute has had the technology to run its jobs on remote machines for many years with SAS/Connect. It utilizes protocols such as TCP/IP to connect to a remote machine and have your program run remotely. SAS Grid computing leverages this along with other software such as the Grid Manager to optimize the performance of SAS on multiple nodes to optimize the computing resources within a grid.



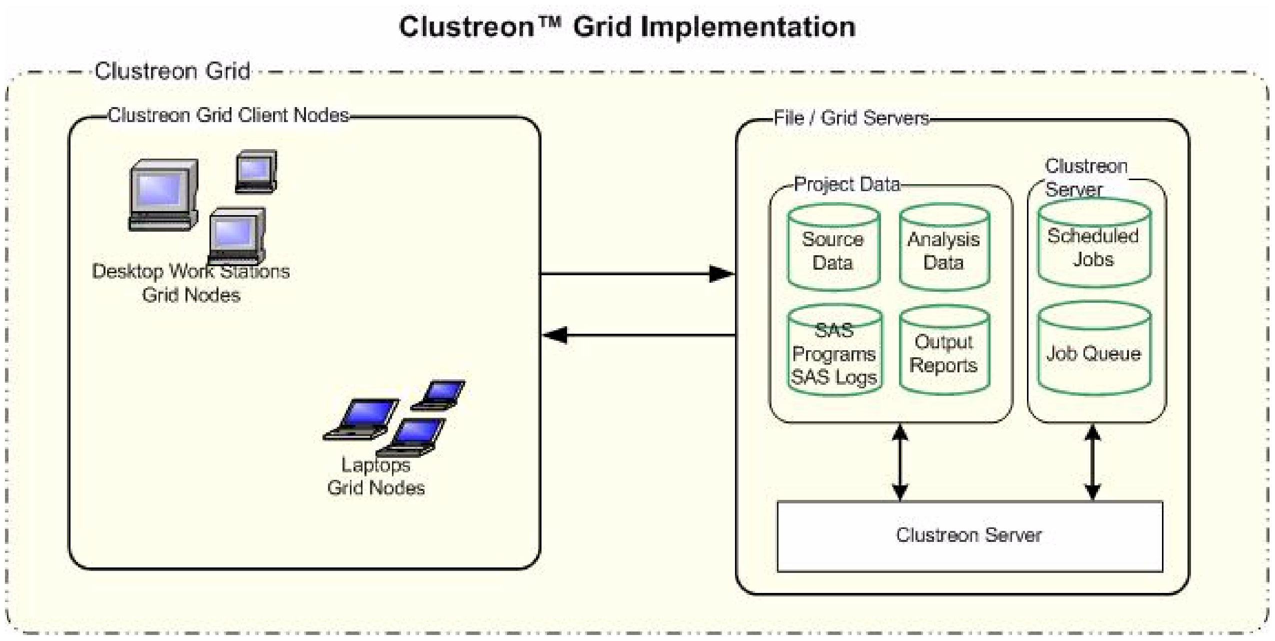
Alternatively, MXI also has a solution Clustreion™ which executes SAS programs within a Grid architecture. This paper will discuss the use of the grid computing environment to help you optimize your computing environment so you can optimize the use of your hardware. In a computing environment of analytics that is resource intensive, it is wise to optimize your performance due to the dynamic environment with diminishing returns on your hardware investments.

GRID ARCHITECTHRE

There are different approaches towards designing a grid computing architecture. Some of the design principles that Clustreon™ accomplishes include the following:

1. **Simplicity –** The design is simple in that there are few components to the system. In a complex system, if there are many distinct components that are dependent upon each other to operate, it is more likely that it will break. Therefore in this case, less is better.
2. **Ease of Administration –** With fewer pieces to the puzzle, the system becomes easier to administer. The users of each node are empowered to have control over various functions of the grid and therefore there is less overall administration.
3. **Dynamic –** There is the ability to add and remove nodes to the Grid without having to cause disruption to the system as well.
4. **Cost Effective –** The software of each piece is simple enough to be less expensive. This is coupled with ease of administration in lower operating costs.

One of the architectural design approaches which accomplishes this is to add intelligence to each node and have the central control be a central database. In this case, the central controller is just SAS tables rather than complex pieces of software. This functions as a form of traffic police while the intelligence and execution of the SAS jobs remain on the client nodes of the grid. The central tables are updated and managed by the nodes which determine the optimal performance of each job. The computing processing will therefore remain spread throughout each node within the grid. The machine that manages the grid controlling tables is tuned for this function and is often a file server. In this design, the file server functions as both a traditional file server and the main grid controller.

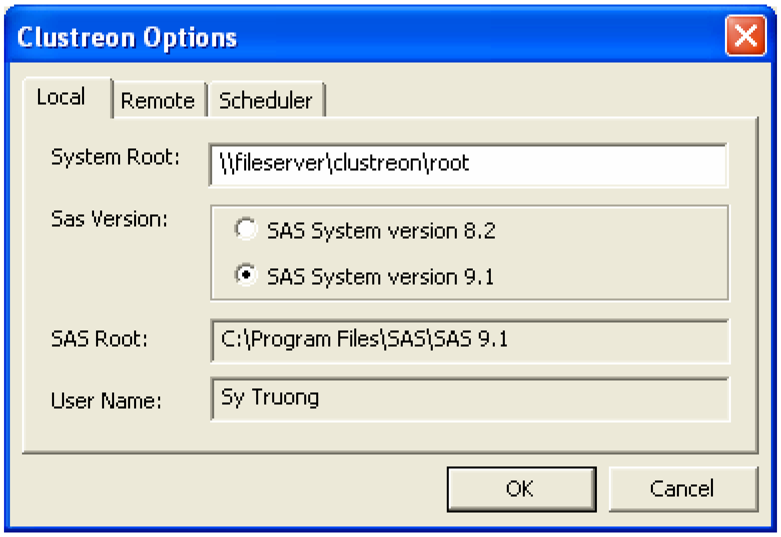


SAS and Non-SAS Nodes

You can add your machine to the node as an active SAS node which will execute SAS programs. Alternatively, you can be a node on the grid without SAS yet still have the ability to assign SAS programs and appoint the processing of SAS to another machine. The configuration of the two types is slightly different. However, they share one thing in common in that they communicate to one central server which manages the processing of SAS programs.

Configuring Local SAS and Remote Node

Within this architecture, you are making your desktop computer available for the SAS processing of the grid. The main setting here is to point your machine to a central system root where all the job queues are stored.


Clustreon Options

Local | Remote | Scheduler

System Root: \\fileserver\clustreon\root

Sas Version: [ ] SAS System version 8.2 [✓ SAS System version 9.1]

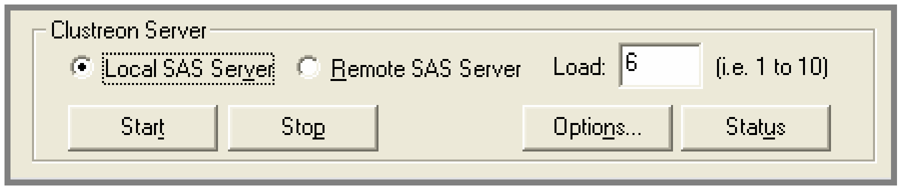
SAS Root: C:\Program Files\SAS\SAS 9.1

User Name: Sy Truong

In this setting, you need to configure the following:

- **System Root –** This describes the central location where the Clustreon™ list of jobs is located. This central database contains all the jobs that are to be executed. If your machine is a node on the server that is designated to process SAS, it will pick up jobs from this central root.
- **SAS Version –** If your machine has multiple versions of SAS, you can select which version to be used to execute the jobs.
- **SAS Root –** This is where it will select the SAS executable. This is determined by the selected SAS version.
- **User Name –** This is the user which you are logged on as and will be the user who executes the jobs. The privileges will therefore be limited to what this user has access to.

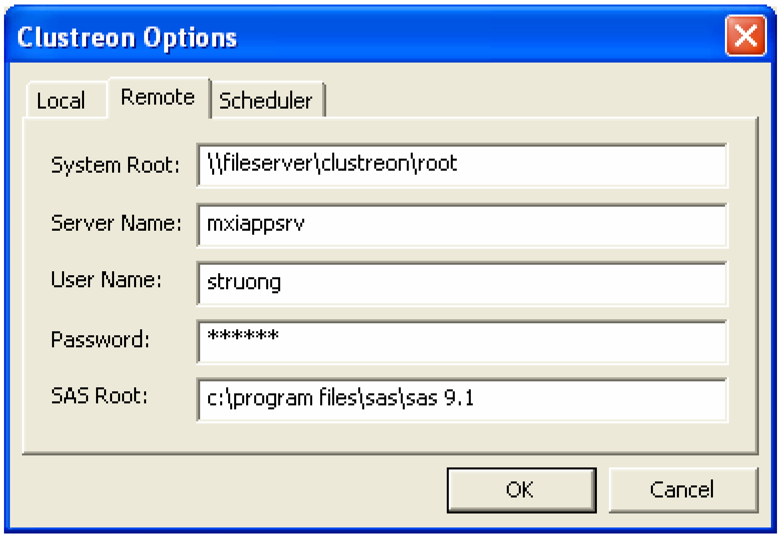
In this case, the processing of SAS will be considered "local" to your machine. The programs and data are located on a different machine, but your machine will do the heavy lifting of processing the SAS programs. You can also control the load in which you want your machine to endure.


Clustreon Server

[✓ Local SAS Server] [ ] Remote SAS Server Load: 6 (i.e. 1 to 10)

The load can range from 1 to 10. If you were to select 10, this means that your machine will take upon up to 10 jobs simultaneously before it will wait for the next job. If you were to choose a smaller number, it would process fewer jobs and therefore will have a smaller load.

If you do not have SAS on your machine but still would like to have the ability to assign jobs, you can configure the "remote" settings.


Clustreon Options

Local | Remote | Scheduler

System Root: \\fileserver\clustreon\root

Server Name: mxiappsrv

User Name: struong

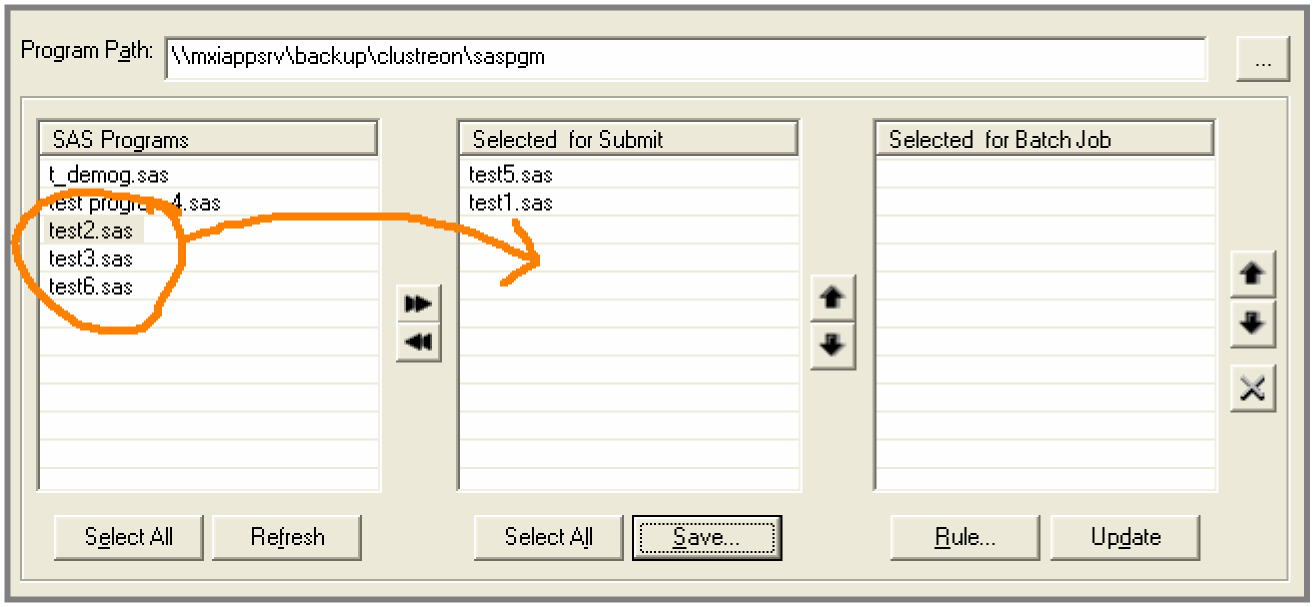
Password: ******

SAS Root: c:\program files\sas\sas 9.1

In this case, if you have permissions, the assigned jobs will be executed on the remote server that you have specified.

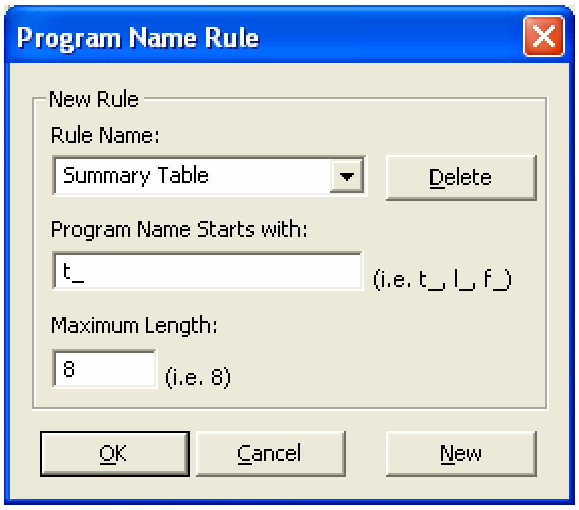
SELECTING SAS JOBS

A SAS job can consist of one or one hundred programs. The order can be significant since a program can produce a dataset that another program uses as input. SAS programs can come from different locations and therefore they can be selected from more than one directory.


Program Path: \\mxiappsrv\backup\clustreon\saspgm

SAS Programs	Selected for Submit	Selected for Batch Job
t_demog.sas	test5.sas	
test program 4.sas	test1.sas	
test2.sas		
test3.sas		
test6.sas		

The user interface provides you with arrow keys or drag and drop to apply your selections. The order can be changed to fit the dependency requirements of your programs. Once all programs have been selected with proper order, you can assign a name to your job. In addition to having groups of programs saved as a job, you can add naming convention rules to ensure consistency.



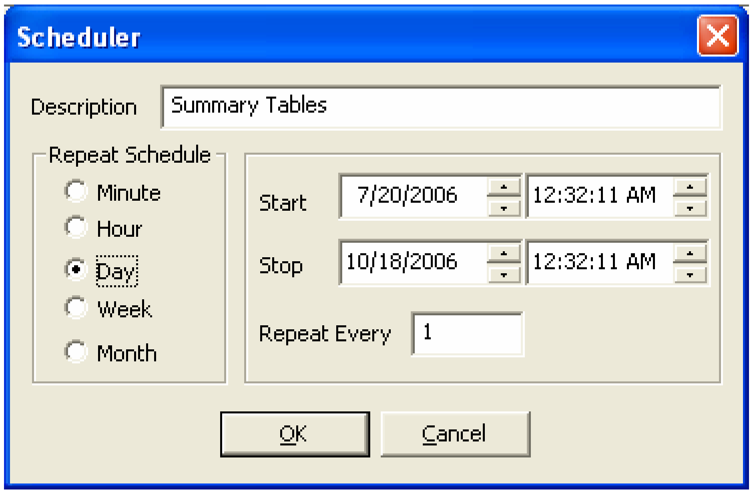
Rule Name: Summary Table

Program Name Starts with: t_ (i.e. t_, l_, f_)

Maximum Length: 8 (i.e. 8)

In this example, the program names must start with the characters "t_" and cannot exceed 8 characters in length. If a rule is broken, the invalid program names will be displayed in red to prompt the user to apply updates accordingly.

Once a job has been assigned, you can schedule this to be executed repeatedly at specified times. This is valuable for jobs that are dependent on scheduled data updates.



Description: Summary Tables

Repeat Schedule: Minute Hour [✓ Day] Week Month

Start 7/20/2006 12:32:11 AM

Stop 10/18/2006 12:32:11 AM

Repeat Every 1

MEASURING SPEED

It is essential to have metrics on performance before you are able to judge if one system is better than the next. There are many forms of measurements since there are many ways of using SAS. The two main classifications of measuring SAS performance include elapsed time and the type of usage.

- **Elapsed Time** – This is a measurement of how long it would take to perform a task. Even within the elapsed time, there are several methods of measurements. These include:
    - **Real Time -** The Real Time represents the elapsed time or "wall clock" time. This is the time spent to execute a job or step. This is the time the user experiences in waiting for the job/step to complete.
    - **User CPU Time -** The time spent by the processor to execute user-written code. This is user-written from the perspective of the operating system and not the customer's language statements. This is all SAS system code that is not operating system code.
    - **System CPU Time -** The time spent by the processor to execute operating system tasks that support user-written code (all CPU tasks that were not executing user-written code). The user CPU time and system CPU time are mutually exclusive.

- **Type of Use –** There are many ways of using SAS. It would therefore be difficult to measure all the ways of using SAS. For our purpose, this was grouped into the two modules of the SAS system including:
  - **BASE –** This includes SAS BASE code which has data step manipulations and many of the SAS function calls that SAS utilizes.
  - **STAT –** This includes the many SAS statistical procedures used to apply its statistical analysis.

SAS has a system option that assists in capturing these metrics named FULLSTIMER. Once this option is turned on, the SAS system will capture the performance statistics as you execute your SAS programs. It does this one data step and one procedure at a time. The results are then listed in the log. An example result for UNIX is shown here:

```
NOTE: DATA statement used:
real time                   0.06 seconds
user cpu time               0.02 seconds
system cpu time             0.00 seconds
Memory                      88k
Page Faults                  10
Page Reclaims                 0
Page Swaps                    0
Voluntary Context Switches   22
Involuntary Context Switches  0
Block Input Operations       10
Block Output Operations      12
```

For each operating system, SAS collects slightly different metrics. However, the main elapsed times are the same. You can use the option for your SAS code to isolate memory issues or I/O bottle necks. It is a useful diagnostic tool. However, when you are measuring different computer systems or methods of computer usage, you need a standard metric and method of measurements. If you do not have a standard way of evaluating performance measurements, it would be like comparing apples to oranges.

Statmark™ Score

Statmark™ is a standard benchmark score that can be applied on different operating systems and different computer system configurations to measure how SAS performs. It utilizes the FULLSTIMER system option to perform its measurements. Instead of measuring customized user programs, however, it executes a standard set of benchmark programs focusing on the two main usage types, BASE and STAT. These programs are based on the IQ/OQ suite of programs that are used to validate the SAS system. It is therefore comprehensive and representative of what the SAS system will do. The final Statmark score derived from the sum and average of the elapsed time for the specified set of standard jobs.

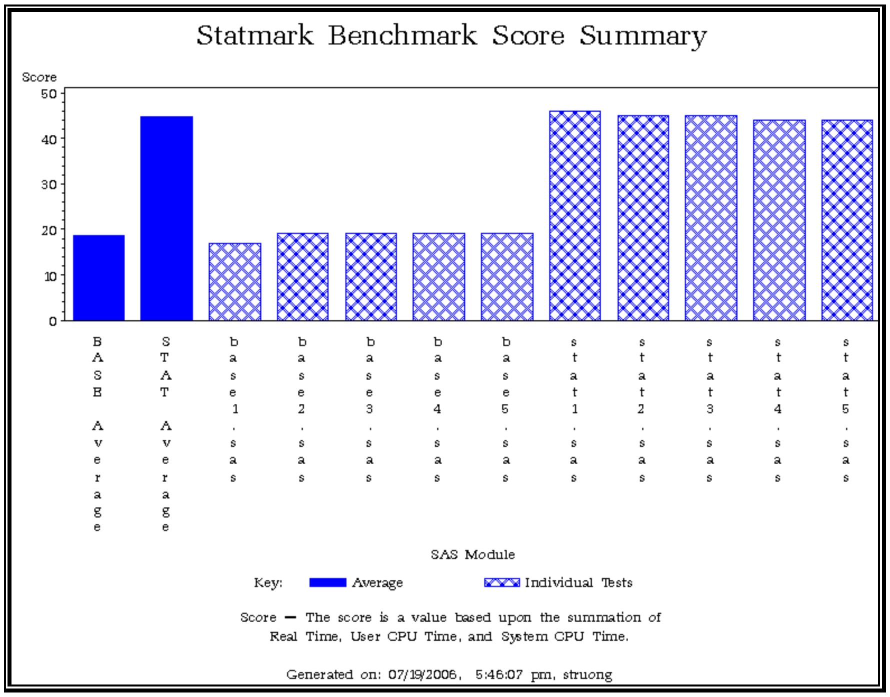
Besides measuring the different types of elapsed time and the types of usage, another dimension is the load that is placed on the system. In the real world of performance qualification, this is usually represented by multiple users on the system running jobs at the same time. The load is therefore simulated in the test by simultaneously submitting multiple SAS jobs. This is an example report that shows a load of 5.

SAS Module	Real Time	User CPU Time	System CPU Time	Memory	Score
base1.sas	0:17:00.78	0:00:00.45	0:00:00.65	2628k	17.03
stat1.sas	0:43:00.14	0:01:00.09	0:02:00.28	2594k	46.01
base2.sas	0:19:00.32	0:00:00.40	0:00:00.65	2628k	19.02
stat2.sas	0:43:00.17	0:01:00.31	0:01:00.73	2594k	45.02
base3.sas	0:19:00.03	0:00:00.53	0:00:00.53	2628k	19.02
stat3.sas	0:42:00.93	0:01:00.23	0:02:00.21	2594k	45.02
base4.sas	0:19:00.68	0:00:00.39	0:00:00.67	2628k	19.03
stat4.sas	0:41:00.90	0:01:00.32	0:02:00.12	2594k	44.02
base5.sas	0:19:00.06	0:00:00.46	0:00:00.65	2628k	19.02
stat5.sas	0:41:00.43	0:01:00.39	0:02:00.15	2594k	44.02
BASE Average	0:18:36.37	0:00:00.45	0:00:00.63	2594k	**18.62**

SAS Module	Real Time	User CPU Time	System CPU Time	Memory	Score
STAT Average	0:42:00.51	0:01:00.27	0:01:48.30	2594k	**44.82**

In this example, the suite of standard BASE and STAT programs were executed five times simultaneously. Each set of jobs has its own elapsed time measured separately and then averaged. The sum of these is what forms the score of 18.6 for BASE and 44.82 for STAT. These two numbers are the scores that can be compared to other systems under the same types of tests.

Another representation of the same information can be displayed graphically as seen here:



Statmark User Interface

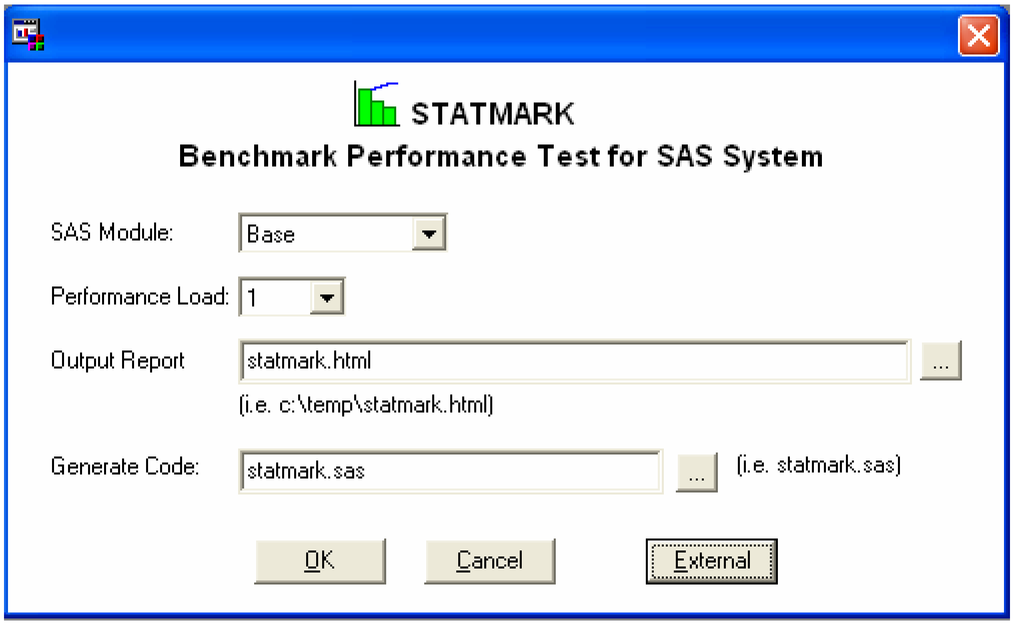
Statmark is delivered with two interfaces to provide flexibility for users. For those in a batch environment, the macro interface is more suitable. In this case, you would type a program such as the one below:

```
%statmark(module=both,
           load=5,
          output=c:\temp\statmark.html);
```

In this example, the selections are:

- **Module** – This specifies either BASE or STAT or both.
- **Load** – This specifies the number of simultaneous jobs that are submitted during the test. Example standard loads include 1, 5, 10, 20, 30, 50 and 100.
- **Output** – The output specifies the location and HTML file which contains the test results.

The complete reference documentation for the %statmark macro can be found at: http://meta-x.com/statmark/statmark.html. In addition to the macro, a graphical user interface is available.


STATMARK

Benchmark Performance Test for SAS System

SAS Module: Base

Performance Load: 1

Output Report statmark.html

(i.e. c:\temp\statmark.html)

Generate Code: statmark.sas (i.e. statmark.sas)

The graphical user interface makes it easier for the user in that you don't have to remember the default parameter values. These are presented to you so it is just a matter of selecting the option of your preference. In addition, there is an option to select "external" logs. If you were to have executed the suite of BASE and STAT scripts, the external option will evaluate those logs and generate the same report. Statmark was developed in SAS/AF but you do not need SAS/AF to run the application. All that is needed is BASE SAS and if you plan to execute the STAT test, you would need STAT.

Statmark for Free

Statmark is available as a free download. You can learn more about this at http://meta-x.com/statmark. It was originally developed for Windows but has been ported to UNIX Solaris. If you have other operating systems, you can download the transport version and execute the program which converts it back to your operating systems format. It is recommended that you run a test on your system and then send your results to info@meta-x.com to be posted. This way, you can see how your system compares to others that have been uploaded.

CONCLUSION

In the world of bloated databases and complex data models, analytical computing is hungry for speed. The evolution of the personal computer has changed the landscape of how enterprise computing is taking shape. The economy of scale and the commoditization of memory, disk storage and CPUs have lowered prices of computer systems while elevating the performance of the most entry level desktops and laptops. Grid computing offers a potent solution in a time that is ripe to take advantage of these forces. You can have a group of PCs function together as a powerful server with minimal cost yet maintain flexibility and scalability. Designing, implementing and managing a grid doesn't have to be a complicated, multi-tiered, expensive undertaking. You can take advantage of free tools such as Statmark to measure your current server performance in comparison to the new grid implementation and see the gains in performance and cost savings. This is coupled with the simple yet elegant solution of Clustreon™ which utilizes existing SAS licenses without requiring additional SAS modules or complex multi-tier computer infrastructure to gain the benefits of Grid computing.

REFERENCES

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Statmark, Clustreon and other **MXI** (Meta-Xceed, Inc.) product names are registered trademarks of Meta-Xceed, Inc. in the USA.

ABOUT THE AUTHOR

Sy Truong is President of MXI (Meta-Xceed, Inc.) They may be contacted at:

Sy Truong

1751 McCarthy Blvd.

Milpitas, CA 95035

(408) 955-9333

sy.truong@meta-x.com