SAS1968-2015

The Truth Behind the Most Comment Myths for SAS Grid Manager

Margaret Crevar, Glenn Horton, Doug Haigh, SAS Institute Inc.

ABSTRACT

SAS® Grid Computing promises many benefits that the SAS® community has been demanding for years, including workload management of SAS applications, a highly available infrastructure, higher resource utilization, flexibility for IT infrastructure, and potentially improved performance of SAS applications. To implement these benefits, you need to have a good definition of what you need and an understanding of what is involved in enabling the SAS tasks to take advantage of all the SAS Grid nodes. In addition to having this understanding of SAS, the underlying hardware infrastructure (cores to storage) must be configured and tuned correctly. This paper discusses the most important things (or misunderstandings) that SAS customers need to know before they deploy SAS® Grid Manager.

INTRODUCTION

SAS Grid Computing can be the basis for a shared, highly available, high-performance analytics environment. However, it is critical that your overall objectives and computing environment be aligned in order for you to achieve success with your SAS Grid Manager implementation and to get the maximum benefit.

This paper explores some of the best practices for setting up a high-performance, high-availability SAS analytics environment. This paper presents these best practices in “myth-busters” format, addressing some of the most common misunderstandings. .

WHEN TO SAY YES TO THE GRID

To start, you should understand the criteria associated with the decisions to implement SAS Grid. In surveying SAS Grid implementations, we have found that the most motivating factors are”

- Your SAS applications are mission critical, and you need to set up a highly available infrastructure.
- You have lots of SAS users running lots of SAS applications, and you want to implement a shared SAS analytic environment that allocates resources as needed.
- You have end-of-month or end-of-quarter SAS processing that has very tight service level agreements (SLAs), and you need to be guaranteed that your computing resources meet these SLAs.
- You would like to establish a hardware and software infrastructure that can be scaled out to meet your ever-growing SAS user base and the ever-growing data that is being analyzed by these SAS users.
- You would like to gain some performance improvements by converting your existing long-running SAS processes to a distributed processing format, by using the new SAS® High Performance Analytics processes, or by using both approaches. Please note that not all SAS processes are good candidates for distributed processing. For example, processes that rely heavily on OLAP processing do not lend themselves to parallelization.

IMPORTANT CONCEPTS – TOO OFTEN MISUNDERSTOOD

Next, we can examine several important concepts you need to consider as you develop your SAS Grid deployment plan. These important concepts are based on direct experience of the SAS Grid R&D team, deployment team and the SAS R&D Performance Lab, and are compiled from working with hundreds of SAS Grid customers. If these topics had been properly addressed at several of our largest SAS Grid implementations, we would have prevented delays in getting these implementations running optimally.

MYTH #1: DISTRIBUTING A SAS APPLICATION ACROSS MULTIPLE MACHINES IN THE SAS GRID WILL MAGICALLY MAKE EXISTING SAS APPLICATIONS RUN FASTER (I.E. THERE IS NO NEED TO ADD MORE IO THROUGHPUT TO THE INFRASTRUCTURE)

SAS customers who are unhappy with the amount of time their SAS jobs take to run on a single SMP system, often assume that moving their SAS jobs to a SAS Grid, with the same storage in place, will result in the distributed features of SAS Grid making their SAS jobs run faster. This might help, but only if you have additional IO capacity. This increased capacity is needed to meet the additional throughput needs that result from the additional simultaneous SAS jobs that can be handled with your existing storage. In many cases, the real reason for slow performance is that the existing IO infrastructure cannot support the single instance of the SAS job that was running. In these cases, additional copies of the job might increase the total distributed run time.

Another topic that is important to understand is what your SAS applications are doing before you distribute them to several concurrent SAS sub-processes. You need to understand what is involved in both splitting the data into multiple buckets and in merging the results. This is important because, if you have too many SAS sub-processes, merging the results back together in the SAS application in order to finish might end up taking longer than expected. Therefore running a SAS job across multiple nodes in the SAS Grid might not produce the fastest results.

MYTH #2: A CLUSTERED FILE SYSTEM ISN'T NEEDED

During hardware setup discussions, we are often asked why a clustered file system is needed with a SAS Grid implementation. Why is a traditional file share not enough? And what exactly does clustered file system mean?

In short, the clustered file system is one of the most critical components of a SAS Grid deployment because it is where all the SAS data, the configuration information, and sometimes the binaries, are centrally located and shared across the machines. It feeds all of the servers with the data that has to be processed. The system you choose should meet these requirements:

- Be reliable
- Be fast
- Be able to handle the concurrency
- Look the same from all machines

A good description of the requirements and possible candidates are found in the paper A Survey of Shared File Systems: Determining the Best Choice for your Distributed Applications.

One of the conclusions from this paper is that clustered file systems provide the best performance.

Another conclusion from the paper is that a Network File System (NFS) is not the same as a clustered file system. To maintain file data consistency, NFS clients invalidate any data stored in the local file cache when it detects a change in a file system attribute. This significantly interrupts the performance of data that is operated on in the local file cache, and this behavior is markedly pronounced on heavy, sequential WRITE activity. This NFS- specific behavior significantly punishes the large block, sequential WRITE-intensive performance used by SAS. If your application workload uses heavy sequential WRITE activity (which is especially true of SAS WORK and SAS UTILLOC file systems), then we typically recommend that you do not use NFS mounted file systems. NFS also does not support file-locking, and it can be problematic for using as a shared file space for permanent SAS data.

Finally, the paper discusses which clustered file systems work best with SAS and why several of the inexpensive file systems do not work well with SAS.

MYTH #3: SAS WORKLOAD IS CONTINUOUSLY RE-BALANCED ACROSS THE SAS GRID NODES

When a SAS job is started, LSF queries the SAS Grid nodes and starts the SAS job on the least busy SAS Grid node. This works great when all the SAS jobs run for about the same amount of time. However, if you have a mixture of SAS jobs, with some running for minutes and others running for hours,

you run the risk of having multiple long-running SAS jobs all on a single SAS Grid node. Note that it is not possible for SAS Grid Manager to know how long a single job will run or to move a running SAS job from one SAS Grid node to another. It is not a problem for multiple SAS jobs to run on a single node unless they collectively start to exhaust resources such as CPU or I/O. In that case SAS Grid Manager provides the ability to set thresholds to automatically suspend some of the workload to enable other jobs to finish more efficiently. One the other jobs finish, the suspended jobs will automatically resume and run to completion.

MYTH #4: SAS GRID MAGICALLY PARALLELIZES EXISTING SAS CODE

One of the features of SAS Grid Manager is the ability for your SAS applications to take advantage of more processors on many different OS instances against a very large storage array. As a result, this feature will make these SAS applications run faster. This is a true statement, but it does not happen automatically by just installing SAS Grid Manager on a cluster of systems. Making SAS applications run faster will require you to add logic to your existing batch SAS jobs to distribute the SAS job across multiple systems. There are also several SAS products that generate code as the result of a user created workflow, such as SAS Enterprise Miner and SAS Data Integration Studio. These products will automatically generate parallel code based on the dependencies in the workflow.

The SCAPROC procedure, added to Base SAS in SAS 9.2, can assist in this process. PROC SCAPROC is an instrumenting procedure that activates hooks within SAS. These hooks allow information to be recorded and subsequently analyzed to help you to parallelize your SAS code. Once the SAS code has been parallelized, you can use this information together with SAS Grid Manager to enable various SAS steps to run in parallel. These changes can improve the execution time of the SAS job, as long as there are enough computing and IO resources to support the extra SAS sessions. The SCAPROC procedure will get you 80% of the way toward parallelizing your code. You have to do the remaining 20%. It is not a one-stop shop for parallelizing SAS code.

These SAS Global Forum papers provide more details about how to use the SCAPROC procedure:

- Thoroughly Modern SAS: The SAS Code Analyzer Helps Bring Programs Up to Date
- Introducing the SAS Code Analyzer

Note that not all SAS job can be distributed, so you need to use the SCAPROC procedure to determine what can and cannot be distributed with legacy SAS jobs.

MYTH #5: SAS GRID QUEUES KNOW WHAT SAS PRODUCTS YOUR EXISTING SAS CODE USES, AND THEY CAN DIRECT JOBS TO THE APPROPRIATE MACHINES WITH NO GUIDANCE FROM USERS

We all know the benefits of using queues to schedule SAS jobs submitted to the SAS Grid. These queues allow you to manage the amount of processing that is sent to each SAS Grid node and to make sure each of these nodes has the resources needed for the SAS job when it is started on the node. You can even use these queues to set priorities on the SAS jobs that are submitted to each SAS Grid node, which enables you are able to achieve any SLA associated with the SAS job.

LSF (which is the application used to support SAS Grid queues) does not scan the SAS code for each SAS job in order to determine what SAS products will be used by that SAS job. Also, by default, the queue manager assumes that all SAS products are installed on all the SAS Grid nodes.

However, there are ways that you can use LSF to create queues to run SAS applications on only certain SAS Grid nodes. You can then educate your SAS users to send their SAS jobs to the specific queues that are used for these tasks. For example, you could define LSF queues for your production SAS jobs or to support your SAS Enterprise Miner users. You can then run these types of SAS jobs on just a few of the SAS Grid nodes.

Any of the following methods can be used to direct SAS jobs to a specific queue:

1. Default grid options in a logical grid server definition, in an application server context
2. Application-specific grid options in a grid options set, in a logical server definition in an application server context
3. On the *grdsvc_enable* function or on the SASGSUB command
4. In the flow properties of a scheduled SAS job (to set the queue for all jobs in the flow)
5. In the advanced properties of a scheduled SAS job (to explicitly set the queue and override any flow properties settings).

If you are interested in more details about these approaches, please stop by the SAS Grid Computing demo station in the SAS Demonstration Area at SAS Global Forum.

MYTH #6: HIGH-AVAILABILITY AND FAILOVER FOR CRITICAL PROCESSES HAPPENS AUTOMATICALLY WHEN YOU INSTALL SAS GRID MANAGER

A very important topic during most SAS Grid workshops is discussing the degree of high-availability and failover that each SAS Grid Manager customer wants from their SAS Grid infrastructure. Part of this discussion centers around the concepts themselves. The other part is around the computing resources needed to accomplish the needed level. After the decisions have been made, additional discussions are required to determine how much of the high-availability and failover capabilities can be accomplished automatically and the extent of additional code changes that might be needed.

To understand the issue, start with the SAS Metadata tier. SAS 9.4 introduced the concept of a clustered SAS Metadata. This cluster requires a minimum of three separate instances of an operating system to support the clustered metadata infrastructure. After installation, no additional SAS coding is required for SAS jobs to take advantage of the failover capabilities of this new SAS feature. However, SAS Grid could be used for restart capabilities of each metadata instance. In addition, you may choose to run a single instance of the SAS Metadata Server and use SAS Grid Manager to provide both failover and restart. The following papers provide much more detail on this topic:

- Best Practices for Implementing High Availability for SAS 9.4
- High Availability Server with SAS Grid Manager

Next, consider the SAS middle tier. SAS 9.4 also introduced the concept of a clustered SAS middle tier. This cluster requires a minimum of two separate instances of an operating system. Like the SAS Metadata cluster, no additional SAS coding is required after installation for SAS jobs to take advantage of the failover capabilities of this new SAS feature. SAS Grid can also be used for restart capabilities as detailed in the paper mentioned above.

Finally, there is the SAS compute tier. On these tiers, SAS jobs are executed on one or more SAS Grid nodes. If one of these SAS Grid nodes were to fail, by default the next step, would be for the SAS user to restart the SAS job (from the beginning) on a different SAS Grid node. However, batch job submission provided by the SAS Grid Manager Client Utility and SAS Schedule Manager, combined with LSF queue policies and the SAS checkpoint restart feature, provides the following support and solutions for this problem:

- The capability to restart a job from the step after the last successful job step
- The ability to set up a special queue to automatically rollover jobs from a failed node to another node in the SAS Grid to continue execution.

For more details about using this restart feature with your SAS jobs, see the topic "Restarting Jobs" in Grid Computing in SAS.

For more details about setting up a high-availability environment, see the topic "High Availability for Critical Applications" in Grid Computing in SAS.

CONCLUSION

SAS Grid Manager is a powerful tool for leveraging multiple systems to run SAS. We have highlighted some of the common misunderstandings to help guide your planning. Considering these topics during your assessment, and discussing them during your SAS Grid Manager workshop, will help to ensure that you have a wonderful SAS Grid Manager experience.

ACKNOWLEDGMENTS

Many thanks to Tony Brown, Cheryl Doninger and Jim Kuell for their review of the content of this paper.

RECOMMENDED READING

SAS Usage Note 42197 contains a list of recommended white papers. Topics include::

- SAS administration
- Operating system tuning
- IO subsystem testing, performance, and tuning
- Shared/clustered file systems
- Testing IO for SAS file systems
- General performance troubleshooting
- SAS Grid environments

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Margaret Crevar
100 SAS Campus Drive
Cary NC 27513
SAS Institute Inc.
Margaret.Crevar@sas.com
http://www.sas.com

Glenn Horton
100 SAS Campus Drive
Cary NC 27513
SAS Institute Inc.
Glenn.Horton@sas.com
http://www.sas.com

Doug Haigh
100 SAS Campus Drive
Cary NC 27513
SAS Institute Inc.
Doug.Haigh@sas.com
http://www.sas.com