Experience with on-demand physics simulations on the Sun Microsystems Computing Facility (SunGrid) at Network.com

J. Lauret[1], **M. Potekhin**
Brookhaven National Laboratory, Upton, NY11973, USA

G. Carcassi, A. Shamash[2], **R. Valia**
Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA95054, USA

jeromel@bnl.gov

Abstract. The simulation program of the STAR experiment at the Relativistic Heavy Ion Collider (Brookhaven National Laboratory) is growing in scope and its responsiveness to the needs of the research community. In addition, there is a significant ongoing R&D activity focused on future upgrades of the STAR detector, which also requires extensive simulations support. In addition to the local computing facility, the Open Science Grid (OSG) resources have been successfully used in STAR. However, the explosive growth of both computational needs and the available computing power, combined with distributed nature of the latter, dictate that all available options are considered – from open source to commercial grids. The computing facility of Sun Microsystems (the SunGrid) aims to deliver enterprise computing power and resources over the Internet, enabling developers, researchers, scientists and businesses to optimize performance and speed time to results without investment in IT infrastructure.

1. Introduction

The Solenoid Tracker At Rhic (STAR)[1] simulation program remains an ever active area of work, catering to the needs of the physics working groups within the STAR collaboration, as well as detector Research and Development (R&D) projects. To achieve the diverse goals, a variety of event generators and detector models are used. As an example, here are a few of the projects handled in 2007:

- STAR detector simulations based on the Herwig[3] and Pythia[4] event generators, in various detector configurations (cross-section and trigger rate studies)
- B^0,B^+ and Hijing[5] event mixing run (event enrichment technique)
- Simulations of the proposed high-resolution pixel-based tracking detector
- Study of characteristics of the proposed muon telescope detector
- Study of non-photonic electrons

[1] To whom any correspondence should be addressed.
[2] To whom any correspondence should be addressed.

STAR generates a few dozen datasets of simulated data per year, containing ~10^7 events. Of these, there are typically ~10^6 heavy-ion events, which require considerable resources for both simulation and subsequent reconstruction, and represent a significant driver of our demand for computing power.

2. Computing environment of the STAR simulation program and its evolution

2.1. Legacy computing environment

Recently, local computing resources available to STAR are becoming increasingly scarce in relation to the amount of data that needs to be processed. This has driven an in-depth consideration on how efficient the resources are being used in support of the physics program. In order to better understand the simulation production paradigm, we will briefly provide some highlights of the "legacy" mode of running in STAR.

First and foremost, the simulation production has to date made use of dedicated centralized disk space (2TB) for short-term storage of the simulated data. The bulk of the data is moved in and out of the mass storage (HPSS) using scripted procedures and the workload management logic is distributed between the binary executable and the driver script submitting jobs to a local resource management system (batch system) on behalf of a single simulation production account. The workload management logic also made extensive use of communication via the centralized file system by keeping and maintaining a set of tokens which distribute the load among the worker nodes. In other words, the tracking of a production series was made through maintaining a file-based bookkeeping. This scheme had several flaws but more noticeably, the sharing of information as per the workflow via a single centralized file under a growth of the number of jobs introduces instabilities over the entire workflow. Additionally, the aggregate IO performed by all jobs targeted toward a single centrally mounted file system (by approximately 0.5k clients) would also introduce I/O bottlenecks experienced by jobs accessing the centralized disk, with ripple effect in other part of the system.

We have come to the conclusion that the old mode of operation does not scale due to saturation of storage, bandwidth and CPU slots under the loads that are currently experienced, and will likely prove inadequate going forward.

2.2. The Grid solution

During our assessment, we concluded that in many cases, it was possible to package all the necessary components, such as the binary executable, shared libraries, and configuration scripts into a self-contained unit (such as a tarball). This way, any dependency on a particular file system for software distribution or re-distribution (such as AFS[5]) can be eliminated. Obviously, the executable and libraries must be compatible with the intended target platform (which often means it is necessary to build the application and its components on the specific platform, for that purpose).

We have tested this approach to run the STAR Monte Carlo production jobs on two different Grid systems, the SunGrid[7] (computing utility developed by Sun Microsystems) and the Open Science Grid (OSG[8]). Our interest in SunGrid comes from the realization of the need to anticipate and understand trends which are leading to the emergence of industrial grids, as well as issues with interfacing and utilization of such novel type of resource. It is indeed conceivable that in future we'll be facing a "grid-of-grids" and the sooner the experimental workflow is self-aware of the breadth of solutions and interface, the better the positioning in terms of exploiting heterogeneous resources. We also note that national Grids such as the China Grid have deployed strategies closer to the industrial grids than of those more light weight developed by initiatives such as the OSG. In addition, such exercise provides useful experience in packaging the payload for execution on remote clusters.

3. SunGrid

3.1. SunGrid basics

The SunGrid is a commercial corporate Grid built by Sun Microsystems. It comprises ~10^3 AMD CPUs located in datacenters owned by Sun, running the Solaris 10 operating system. At the time of our testing, the Web interface was the only officially supported interface of the SunGrid, available through their portal. However, based on our request, Sun had also provided a beta version of a standalone API.

In the SunGrid pricing scheme, users are billed a flat rate of $1 per CPU-hour. Given the nascent nature of the open market of farmed CPU, the price does not yet reflect the economic realities of our experimental community. Users can be allotted dedicated storage space on SunGrid, with the pricing scheme being worked out. The STAR Collaboration was granted a block of promotional CPU hours to be able to complete the feasibility study with real production jobs, and also received about 0.5TB of dedicated storage.

3.2. Running jobs on SunGrid

3.2.1. SunGrid resources

In SunGrid's terminology, the various components of the end user's software system are termed "resources". Physically, they are packaged and transferred from the user's machine to SunGrid as compressed (tarball files). Typically, they are given mnemonic names as per the user's choice, and can be browsed (and, to a degree, edited) via the Web interface. There are no set rules as to what specifically can reside inside a SunGrid resource.

3.2.2. Preparing the code

Getting the STAR software to run on SunGrid involved a port from Linux to Solaris 10, which was not always as straightforward as one would like (however entirely doable). The issues encountered involved determining the correct optimization levels for the compiler we chose to utilize, features of the linker and other such utilities.

A simple driver script was created which allowed running a class of jobs by changing the script's input parameters (see below). The paths to various configuration and data files were modified such that the directory structure was essentially flat (i.e. all components were pretty much in the same directory as the executable itself).

The components which were guaranteed to remain static for a long period of time, such as a few libraries and the binary executable itself, were all packaged into a single resource (see above) and uploaded once.

3.2.3. Preparing the data

Input files, produced with the Hijing event generator using local resources, were packaged into a series of SunGrid resources and uploaded to the SunGrid facility under descriptive names. The user interface allows to add comments to each resource in the browser, which somewhat facilitates the procedure.

3.2.4. The SunGrid Job and Run definition

The user must create a job definition, which contains a list of resources that need to be utilized by the job, as well as the script or a binary to be executed at submission time. Input parameters for that script or binary can also be specified inside every job definition, which provide additional flexibility. This way, one can run a variety of job of the same class, without the need to upload or otherwise modify any of the user resources.

A job thus formulated can then be submitted using the Web interface and at this point becomes a "run". Resources, jobs and run status are all monitored via the Web interface as well. The data files produced within a run become separate resources, also traceable and listed in the Web interface. Once they are ready, the data contained in the resources are downloaded back to the user's machine via https, in compressed (tarball) format.

4. Experience of the Star Collaboration with running production jobs on SunGrid

We have successfully utilized SunGrid to run the Hijing/GEANT simulation for the STAR Tracking Upgrade R&D Project, which amounted to ~10^4 heavy ion events. The calculated job efficiency was at its maximum (i.e. 100% success within our sample) although the use of portals may hide job re-submission upon problems. Even though this is a relatively modest workload, the opportunity to use SunGrid opened just at the time when the local resources were oversubscribed by other urgent requests. The simulated data was downloaded from the SunGrid server via https and subsequently used in the performance analysis of the proposed high-precision tracking detector.

5. Conclusions

The STAR collaboration is facing a growing demand in requests for simulations both in diversity and level of complexity. The "old" mode of operation, whereby load sharing was done by the driver script and the executable, and the jobs were submitted to a local farm, did not scale to meet the new challenges and forced to reconsider the approach. As a consequence, a light weight approach using self-described archive whereas all components are included have been used and successfully tested on multiple Grid environments and namely, the resources from OpenScienceGrid consortium and the SunGrid facility. As the model has showed to scale well, running simulation via Grid has been the main (and sole) way for running simulation in STAR ever since. We found that utilizing SunGrid in its current implementation and in a production environment have both several drawbacks and positive characteristics.

A current major drawback is the nature of our community needs, essentially driven by the size of the datasets (Data Grid). The data transfer to and from SunGrid, with its reliance on the http protocol and related firewall and bandwidth issues, is clearly the weak link in the system (however, there are indications from Sun that this is being addressed). At the time of our testing, we were not provided with a programmatic interface to the Sun Computing Utility. This would significantly limit the usefulness of the system in the Particle and Nuclear Physics context. However, again, this since has been addressed by the Sun developers and further progress can be expected.

On the positives, there is no custom client software to be installed on the user's system, therefore one can jump right into the action. In most cases, it is reasonable to expect that the port to Solaris 10, from other Unix-like operating systems, should be reasonably straightforward and hence, the initial workforce investment is well leveraged by the resources potentials at hand. Furthermore, we found that the Web user interface (portal) is clean an unambiguous and that access to dedicated storage space on the SunGrid facility greatly improves the management of produced data and saves a lot of upload time, in the context of simulations. Finally, our interactions with the support Staff of the utility revealed there is a healthy level of support for end users and developers.

References

[1] The STAR collaboration, http://www.star.bnl.gov/

[2] A reference

This reference has two entries but the second one is not numbered (it uses the 'Reference (no number)' style.

[3] The Herwig Event generator, G. Marchesini, B.R. Webber, G. Abbiendi, I.G. Knowles, M.H. Seymour and L. Stanco, Computer Phys. Commun. 67 (1992) 465.

[4] The Pythia Event generator - M. L. Mangano & T. J. Stelzer, Ann. Rev. Nucl. Part. Sci. 55, 555 (2005)

[5] The HIJIING Monte Carlo model, http://www-nsdth.lbl.gov/~xnwang/hijing/

[6] The Andrew File System, OpenAFS dsitribution http://www.openafs.org/

[7] SunGrid fomcpute utility, http://www.sun.com/service/sungrid/

[8] The OpenScienceGrid consortium, http://www.opensciencegrid.org/

[9] China National Grid http://www.cngrid.org/