

Cluster Evolution at the Genome Sciences Centre

Bernard Li, Mark Mayo, Asim Siddiqui and Steven Jones, BC Genome Sciences Centre, BC Cancer Agency

OSCAR Symposium 2004, Winnipeg, Canada, May 2004

Abstract

High Performance Computing has become a leading technology in the field of Bioinformatics. This is partly due to the completion of the Human Genome Project and many various other genome projects around the globe. These projects generated a lot of genetic data, which when mined, could improve our understanding of the many mysteries of biology and diseases like cancer.

OSCAR has helped us de-mystify cluster deployment and management and its collection of leading clustering technologies has greatly enhanced our ability to harness our resource to generate and analyze data at an efficient rate. This paper describes how we have progressed from a relatively simple home-grown cluster environment, to one where compute jobs are better scheduled and managed. Our experience with package creation for OSCAR will also be discussed.

1. Introduction

Our first cluster composed of a hundred IBM xSeries 330 Pentium III compute nodes connected to each other via 100MBps ethernet network. They were named *xofy* (where *x* and *y* are numbers from 0-9) to facilitate displaying of monitoring information in a matrix [1]. These compute notes were accessible for users via remote shell and jobs were executed on them ad hoc with no batch scheduling system.

Demand for the cluster grew and more and more users require CPU cycles to do their analyses. We ran into the situation where users needed to manually log into each node to find the first available node that is not busy to run their jobs. Many nodes were overloaded with jobs whereby the machine would continuously memory swap – it was then that we realized a better system is necessary for managing and maintaining a cluster, so that we can increase the efficiency and usability of the resource.

At that time we were looking at expanding our cluster capacity. With the Opterons hot off the manufacturing lines, we have heard many good things about this new chip from AMD, which can run 64-bit code and not take a penalty in running 32-bit code at the same time [2]. It was an easy decision to make in terms of the architecture to acquire so we started looking at different vendors to provide us with the hardware. Initially we were looking at a turnkey solution from LinuxNetworx. Their management software was quite powerful and provided many advantages over our existing setup [3]; however, the software came with a great premium and at the end we decided to buy more cluster nodes for the same amount of money and this started our quest for a good open source cluster management framework.

2. The Beginning

We tend to execute many short-running jobs on the cluster – a perl script was written to use PVM to automatically look for the least-loaded nodes and then execute jobs from a batchlist [4]. This was our first step in our cluster 'evolution'. Next we also evaluated PBSPro, the commercial version of the popular PBS batch scheduling system [5]. We then realized that although we have a batch scheduling system, we are still lacking a lot of cluster programming libraries, as well as other monitoring/administration tools for cluster management. We then looked at evaluating open source cluster solutions like ROCKS and OSCAR. Each ROCKS release is based on a single version of RedHat and is almost like a Linux distribution by itself [6]. OSCAR, on the other hand can be installed on top of a working 'workstation' installation of selected versions of RedHat and Mandrake. This flexibility is one of the reasons why we picked OSCAR over ROCKS because we would like to install the same version of RedHat on our user's workstations as well as on our cluster. Another appealing advantage we see in OSCAR is that it is modular [7] – most packages outside the core packages are not inter-dependent and you can

choose which ones to install. The idea of OSCAR packages also makes it easy for users to create their own packages that plugs directly into the main system.

3. Present

Currently we have a 134-node cluster, composed of IBM x330s (dual Pentium III 1.4GHz), IBM BladeCenter (dual Pentium Xeon 2.4GHz) and Appro 2121H (dual AMD Opteron 246). We have also ordered seventy more Opteron-based nodes from Sun (v20z, dual AMD Opteron 244) – this will bring us over 200 nodes, making it one of the largest clusters in western Canada¹. We are using commodity ethernet interconnects for our cluster. The x330's are wired together via a 96-port HP 100Mbps Procurve switch. The BladeCenter has a built-in D-Link gigabit switch which provides networking for the 14 blades in the 7U chassis. Two 24-port Cisco 3750 gigabit switches support our 40 Opteron nodes. These are all interconnected together via our redundant Cisco Catalyst 4500 core routers.

Console access to these nodes varies between the three different architectures. The x330's have a special KVM interconnect which can be daisy-chained together. The BladeCenter has a web-based management unit which also provides console re-direction via a Java web-application. The Opterons come with a Cyclades Terminal Server so users can telnet into the server to gain access to the console.

In terms of software, we are running OSCAR version 3.0. We also use Ganglia version 2.5.6 for monitoring our cluster load/usage [8] and for our resource management, we use Sun Grid Engine version 5.3p5 [9]. Other commonly used programs which came bundled with OSCAR includes C3, LAM, PVM, etc.

The home directories of our users reside on a NetApp FAS960 filer NAS (Network Attached Storage) box and is NFS mounted on each node (which deviates slightly from how the default OSCAR installation sets up user home directories). A cluster-wide scratch space is available via */scratch*, which is mounted on each client node. This is where users can store their temporary data files.

¹ In the TOP500 List 11/2003, University of Calgary has a 278-node cluster at #256 and Westgrid has a 1008-node cluster at #58.

Management has improved immensely since the switch over to OSCAR – with tools like C3 [10], it is now very easy to update mount points, hosts entries and other cluster-wide settings that may change from time to time. Previously this was done manually on each machine, now this can be done by editing one master file and then pushing them out to all the client nodes. Job submission is also very simple - users only need to log into the headnode to submit jobs. Jobs are then queued and will run when resources are available. Users can be notified via email as jobs complete saving them a lot of wait time – "Submit and forget it".

4. Cluster jobs specification

Typical jobs that we run at the centre include BLAST, gene prediction, physical map assembly, sequence assembly, regulatory elements predictions and other analyses. Previously a lot of users simply run the analyses on their own workstations. This changed when we introduced the newly deployed OSCAR cluster. Users were encouraged to run intensive jobs on the cluster and also to migrate production pipeline analyses onto it.

Users have found the new system very easy to use and efficient. In fact, our job efficiency has climbed by over 200% with the new system. With a Share-Tree Policy in place with Grid Engine [11], the system is able to schedule jobs fairly. Users who have submitted many jobs in a short burst will be penalized so that jobs belonging to other users can be interleaved in the queues and therefore executed without having to wait for the entire batch of submitted jobs to finish. So far this simple policy has worked really well with our computation demands. Grid Engine is also capable of more advanced features that could for example balance cluster use based on project or department.

As mentioned previously, we generate a lot of short-running jobs (a few minutes on a Pentium 4-2.4GHz CPU) and to facilitate the submission of this large amount of jobs, we have written a home-grown script for automatically creating job scripts for submission to SGE [12]. This script is written in python and uses the SGE concept of an 'array job' [11], which treats each separate computation as a 'subtask' of a job – therefore it is much easier for bookkeeping as all the related jobs have one job identifier and name.

5. Creating packages for OSCAR

An OSCAR package is composed of the RPM package(s) of the program and additional scripts for configuring and installing in an OSCAR environment [7]. Here we will outline some packages that we have in development. Packages are usually quite straightforward to create – it becomes more complicated if the program does not readily come in the form of RPM(s) – in which case RPM package(s) have to be built (for RedHat, Mandrake, etc.) prior to packaging.

Package creation was facilitated by the availability of the OSCAR Package HOWTO which is a document put together by the OSCAR Core Team as a guideline on how to create an OSCAR-compliant package. This document is available at <http://www.csm.ornl.gov/~naughton/oscar/oscar/pkg-howto.pdf> or at the SourceForge CVS tree. This should be the first document to consult if users are interested in creating their own packages. This document along with discussions at the OSCAR users/development mailing-lists assisted us greatly in creating packages.

Ganglia Package

Ganglia is a cluster monitoring tool developed at Berkeley and is widely adopted in many organizations for monitoring the load, memory and network usage of compute nodes [8]. It works with a wide range of different platforms including Linux, FreeBSD, Sun Solaris, among others.

We first started using OSCAR when it was at version 2.2.1 - at that time the Ganglia package for OSCAR was not compatible with the current release. We therefore decided to contribute back to the community and create a new package for OSCAR. The only roadblock we had was that the official RPM would start the monitoring daemon immediately after installation and this prevents installation on client nodes. It was therefore necessary to rebuild the RPM with the auto-starting/stopping of the daemon disabled. In April 2004, the Ganglia Package for OSCAR was released and is fully compatible with OSCAR 3.0 and RedHat 9.0. The package is based on Ganglia version 2.5.6 is now available for download from OPD (OSCAR Package Downloader). This package works with a brand

new cluster installation or a production cluster that has already been deployed.

Sun Grid Engine Package

Currently we are working on our next package for OSCAR, which is the batch scheduling system from Sun. This is an open source project sponsored by Sun [9]. Unlike commercial batch scheduling systems like LSF and PBS Pro, Sun Grid Engine is freely downloadable from the project website. Grid Engine comes in two different flavours – Sun Grid Engine is the standard version and Sun Grid Engine Enterprise Edition contains more advanced scheduling policies. Both are from the same CVS tree and can be used free of charge. It is also possible to purchase a support contract from Sun for the Enterprise Edition. It should be pointed out that the mailing-lists are also an excellent alternate source for assistance. The lists are frequented by the developers and other keen users who are willing to volunteer their time to assist new users with their problems.

We chose to run Sun Grid Engine over the bundled versions of openPBS from OSCAR because we had some stability issues with the openPBS/MAUI combination. So far we have been really satisfied with the performance of SGE under the OSCAR system and we decided to package SGE so that other users can benefit from it. There are now more than one batch scheduling systems for users to choose from. Other than the default openPBS/MAUI package, there is also Torque which used to be called Scalable PBS (from the folks who brought us MAUI) [13] and soon, Sun Grid Engine.

The creation of the SGE Package was more involved than the Ganglia package because there is no official RPM package available from the project site. We therefore had to start from a third-party RPM package [14] and put in some modifications so that it works with the OSCAR setup. The package is currently work in progress and should be released by the end of June, 2004.

One advantage of writing packages for OSCAR is that you can make certain assumptions about an 'OSCAR cluster' and set the default settings accordingly. This makes it very easy for the end-users of the package to install and set up such a package. This is one of the reasons why there is a growing interest from users to use

OSCAR for cluster deployment and management.

OpenLDAP Package

Another future contribution we would like to investigate is integrating OpenLDAP user authentication [15] with OSCAR. Currently, OSCAR uses OPIUM which works by syncing up user information files (passwd and shadow files) between nodes. If OSCAR supports user authentication with a LDAP server, then there is no longer a need to sync the files within the cluster. This is an added value for sites that have a LDAP infrastructure and simply wants to grant cluster access to a subset to their user base.

Bioinformatics Packages

We are also interested in providing some bioinformatics tools for OSCAR users. This has the potential of making OSCAR more competitive in the Bioinformatics HPC world. If tools are readily packaged for use with OSCAR, then scientists can simply select a few analysis tools to be installed so right after the deployment of a cluster, computations can commence immediately.

We have already done some work with integrating bioinformatics software with our cluster and these could serve as basis for future packaging efforts. There are two applications that we have successfully 'ported' to our new system. FPC is an interactive physical map assembly program [16]. The original program was single-threaded and our bioinformatics team supplied a patch to make the application run in parallel using a client/server model [17]. This worked with our old system where users were allowed to fork out jobs to the cluster via remote shell – now with the more controlled batch scheduling system, we need to do some more work to integrate it with the submission system.

With the help of SGE's built-in parallel environment [11], we were able to create our own FPC parallel environment which would start the daemons on the allocated nodes when a FPC job is scheduled and executed – physical map assembly is then performed and when the job is completed, the daemons are automatically shut down by the parallel system. This works well as SGE is totally involved with the scheduling and handing out of resources to jobs.

Another application which we have integrated is Paracel BLAST [18]. BLAST is the leading bioinformatics tool for performing genetic sequence (DNA and protein) alignment and similarity searches [19]. Paracel BLAST is a commercial version of BLAST developed by Paracel which runs the BLAST algorithm in parallel by breaking up the sequence database and/or input sequence and then running the comparisons using multiple daemon processes.

Paracel generally only sell their software bundled with their cluster hardware, as a result their software was written with certain assumptions about the cluster system. Therefore making it work with our cluster infrastructure was quite challenging. At the time of writing, Paracel BLAST has its own scheduling system for managing internal jobs, and their integration with SGE could only ensure that SGE jobs do not interfere with Paracel BLAST jobs (so SGE is actually not involved in resource management) [20]. This is not very effective if we anticipate a lot of other running jobs in SGE. We therefore researched our own way of integration.

We used the same approach as FPC and were able to reproduce a parallel environment in SGE. However because of the fact that Paracel BLAST requires separate licenses to run simultaneously (and also different ports), our current integration can only support one Paracel BLAST instance at a time. Developers at Paracel are working on a more compatible integration with SGE, possibly embedding Grid Engine within the software so that it is SGE-aware.

6. OSCAR-specific issues

There are two main issues we have with OSCAR right now and hope that they can be addressed with future developments with the system.

Upgrading OSCAR

Currently, there is no upgrade path for OSCAR. Basically if you have an older version of OSCAR running on your cluster and you want to use a newer version, the only option is to start the installation from scratch. This is still feasible for a small-scaled cluster (< 100 nodes) but when working with a bigger cluster it may not be possible to bring down the cluster from production simply to upgrade. This is a potential hindrance for mass adoption of the OSCAR system because most users will start with

whatever is the latest release of OSCAR and then not be able to upgrade due to large overhead.

New Distribution Support

Another concern is support for more Linux distributions. RedHat is one of the most supported distributions for OSCAR and with the announced end-of-life for all of RedHat's existing free workstation Linux releases [21], it may be time to add more to the list of supported Linux distributions. Currently Mandrake is supported and there is also a development version of OSCAR that works with Debian IA-64 [22]. Work is underway in porting OSCAR to Fedora as well – this could also pave the way for RedHat Enterprise Linux ES/WS releases.

7. Future Work

With the adoption of the OSCAR cluster management system, setting up and maintaining a cluster has become a relatively simple task. Together with the stability and robustness of Sun Grid Engine, our cluster has improved in performance and uptime. This saves us a lot of time with maintenance and administration so we can spend more effort in developing cluster applications. We are currently looking into mpiJava [23] so that we can combine our wide array of Java-based bioinformatics applications with the Message Passing Interface to harness our cluster resource.

Embarrassingly parallel jobs will probably still dominate our job queues in the next few years – given the fact that our analysis pipelines involve running many instances of the same program on different inputs and datasets. However, with the parallel programming libraries in place, our developers will definitely start to write more parallel algorithms with MPI and PVM in the near future.

A few bioinformatics research centres in the Vancouver area are interested in linking together as a grid to support our combined computation needs. NetCache and other content caching technologies [24] will most likely be used and this will be another area of research we will be participating in. It would also be interesting to tie this together with OSCAR to create grid-enabled OSCAR systems.

8. Conclusion

So far OSCAR has played an important role in the development of our cluster infrastructure. With the new system, both users and administrators find the cluster to be easy to deal with. We anticipate that we will continue to use OSCAR as our cluster management/deployment system and will contribute as much as we can to the HPC community.

We hope our experience can convey to the audience that setting up a cluster is no longer rocket science thanks to open source projects like OSCAR. OSCAR has made cluster installation easy and its applications in the life sciences and computational sciences continue to grow every day. Hopefully more users will join the community and help create more useful packages or help with development/testing of the project.

References

1. CLUSTERPUNCH, a distributed mini-benchmark system for clusters, <http://mkweb.bcgsc.ca/clusterpunch>.
2. AMD Opteron Server Processor, http://www.amd.com/us-en/Processors/ProductInformation/0,,30_118_8796,00.html.
3. Total Cluster Management Software, <http://www.linuxnetworx.com/clusterworx>.
4. Shawn Rusaw. Using `pvm_batch.pl` to Distribute Jobs to Cluster Nodes.
5. PBS Pro, <http://www.pbspro.com>.
6. Rocks Cluster Distribution, <http://www.rocksclusters.org/Rocks/>.
7. Benoît des Ligneris, Stephen Scott, Thomas Naughton, Neil Gorsuch. Open Source Cluster Application Resources (OSCAR) : design, implementation and interest for the [computer] scientific community. *OSCAR 2003 Symposium*, May 11-14, 2003.
8. ganglia, <http://ganglia.sourceforge.net/>.
9. Sun Grid Engine, <http://gridengine.sunsource.net>.
10. Cluster Command & Control (C3) power tools, <http://www.csm.ornl.gov/torc/C3>.
11. Sun Grid Engine 5.3, Administration and User's Guide.
12. Bernard Li. Using `mqsbatch` to submit multiple jobs to SGE.

13. Torque Resource Manager,
<http://supercluster.org/torque/>.
14. Scalable Systems,
<http://www.scalablesys.com/community.html>.
15. OpenLDAP, <http://www.openldap.org>.
16. FPC,
<http://www.genome.arizona.edu/software/fpc/>.
17. Ness, S.R., Terpstra, W., Krzywinski, M., Marra M., Jones, A., and Ness, S.J.M. (2002). Assembly of fingerprint contigs: parallelized FPC. *Bioinformatics* 18:484-485.
18. Paracel BLAST,
<http://www.paracel.com>.
19. NCBI BLAST,
<http://www.ncbi.nih.gov/BLAST/>.
20. Paracel BLAST User Manual Revision 1.5.6.
21. redhat.com | End of Life Products,
<http://www.redhat.com/security/archives.html>.
22. Adam Lazur. OSCAR-Debian,
<http://hackers.progeny.com/~laz/oscar-debian/>.
23. mpiJava,
<http://www.hpjava.org/mpiJava.html>.
24. Network Appliance – NetCache,
<http://www.netapp.com/products/netcache/>.