Resource Management on a Mixed Processor Linux Cluster

Haibo Wang

Mississippi Center for Supercomputing Research

Many existing clusters were built as a small test-bed for small group of users and then upgraded to a large one. The process of building such test-bed is also the process of learning. After finding the applications or user group that is beneficial from such technology, it is time to upgrade the cluster from a test-bed to the production system.

There are two options to upgrade a cluster: adding the same hardware (previous hardware configuration) to existing cluster or adding the current available hardware (node) to the existing cluster. Most likely, the second option is more favorable due to two reasons: the vendor might have a very attractive price on the current hardware and might discontinue the old model; the current hardware will make the cluster more attractive to the users. So a mixed processor cluster is a better choice. The resource management on such heterogeneous cluster will be more complicated than the homogeneous cluster.

MCSR Mixed Processor Cluster

Our cluster started with 16 Pentium III processors as a test-bed which we trained ourselves and our users for the new computing platform. After the positive feedback from our users, we added 40 Pentium IV processors to the cluster and used 3 Pentium IV processors as dedicated I/O nodes. The applications running on MCSR cluster include: MPICH, Linda, NWCHEM[1,2], Gaussian, GAMESS and MPQC. There are about 50 research accounts and a dozen class accounts.

I/O management

Many scientific computing applications are I/O intensive and often need to access large amounts of data stored in files. Some of them need the large scratch disk space such as GAMESS, NWHCEM and Gaussian. Some of them need check pointing for later restart or data analysis such as Gaussian. It is generally known that I/O is much slower than computation. The slow I/O speed is the bottleneck of overall performance. Improving the I/O performance can reduce the overall computational time. The balance among I/O performance, communication performance and CPU performance will also improve the overall performance. It can improve the I/O performance if the I/O operation always access large amounts of data instead of small amounts of data. To improve the I/O of the cluster, we can choose sufficient amount of high-speed I/O hardware and appropriate file system software.

Most of current disk is 10,000 RPM and it takes 6 milliseconds for the disk to complete one revolution. SCSI disk is faster than IDE but cost more. Many ready-to-sell PCs are installed with IDE disks and SCSI disks usually come with server system. The file system

cache with size of 512k is the common configuration in Pentium III/IV machine. Cache can reduce the number of I/O operation and improve the performance.

For a small group or lab cluster, how to choose I/O configuration is dependent on the application running on the cluster. Usually, there are few applications running on the small cluster. It is not cost efficient to have a dedicated I/O node. Many small clusters choose system and I/O combined node to handle I/O request. Each compute node also has a local disk of its own and is used to store scratch files local to each process.

In contrast, it is very important to a large cluster on how to choose I/O configuration in the multiple users environment. We used the following approaches to improve I/O performance.

1: category the user according to two types: researcher who needs the resource to run the job and general user who want to learn the technology. Researcher will get the faster processors and more resources. General user will be assigned to the slower processors and limited resources.

With PBS Pro, we can group the nodes into different queues. Group the slower nodes to a queue that is assigned to the general user such as class accounts. Since PBS assigns the job to the free node according to a nodes file, it is suitable to put the best hardware on the top of lists. The best processor will always be busy if not all the nodes are busy. Research account can access to the faster processor at first. The security and Access control features in PBS Pro can permit the system administrator to allow or deny access on a per-system, per-group, and/or per-user basis.

2: category the job types certain user might run. Some users prefer computing intensive jobs and some prefer I/O intensive jobs. We split the users who are in the same group and run the similar jobs into different I/O nodes. It will help to distribute the load.

3: configure the I/O nodes to handle more workload. We use RAID[3] (Redundant Array of Inexpensive Disks) to combine multiple inexpensive disks into a large, high performance logical disk called work on the I/O node. We assign 10 research accounts and 6 to 8 general accounts to each I/O node. Then we create a virtual directory called ptmp to combine all I/O nodes. Using RAID, we can stripe data across the multiple disks on each I/O node and get high I/O rate by accessing data in parallel.

4: take advantage of the I/O implementation of the applications. For example, user can use MPI-IO implementation to optimize their programs. We have developed online document to educate our users and helped our user to adopt these tools. Another application NWCHEM uses Global Arrays (GA) to support message passing and shared memory by allow task-parallel access to distributed matrices.

As mentioned above, the cache of file systems can help to reduce the number of small I/O requests. But still, for each I/O request, there are system calls even though the system reads large data with the help of file system cache. The cost of these system calls is very high. For the mixed processor cluster, it is very important to make large request.

The ratio of dedicated I/O and computing nodes is related the user/job. On our cluster the ratio is 1 to 17 and it is almost the same ratio of user per I/O node. Since the number of users who prefer I/O intensive job is increasing, we will increase the number of I/O node.

Job Scheduling

Due to the drawbacks of a RAID system such as high disk failure rate and lower throughput of small writes, it requires a complicated job scheduling system. There are several jobs scheduling software available to the cluster system. We installed OpenPBS on the 16 nodes cluster. It works well on the small uniform processor system. When the 16 nodes cluster was upgraded to 54 nodes, OpenPBS cannot provide the complicated job scheduling and user grouping functionality we need. These functionalities can be provided from PBS Pro. PBS Pro provides the capabilities of grouping nodes/users or associate nodes with queue, assigning priority score to user/group at queue or system level, deleting jobs running on dead nodes and preemptive job scheduling.

Using PBS Pro, we can limit the number of queued jobs per user and the number of running jobs/processes per user. It is very important to keep the balance among the I/O node. If we allow user run a large number of jobs, there will be so many I/O processes on certain I/O node and crash the node since the user is bound to the node.

Besides using PBS Pro, we developed some scripts to simplify the task of job submission and keep the nodes in good shape. For example, if a user request more memory than the physical memory available, there will cause a lot of page swapping. Some of the nodes will be very slow to respond to PBS. The performance of the whole system will be decreased.

PVFS[4] is a parallel file system and stripes files across the local disks of nodes in the cluster. It is virtually a single image file system. To use PVFS, we need a more balanced system. It might require all nodes running the same application and with the same resource. In the multi-application multi-user cluster environment, there are some limitations on this approach and we are still testing it. In the small cluster with single application, it is a good choice and is proved to be very efficient.

Kernel Parameter Tuning

We found that increase the value of SHMMAX would help some computing intensive job such as MPI and NWCHEM. **SHMMAX** Is the maximum size of a shared memory segment. The range of values is between 131072 and 2147483647 bytes; the default value is 524288 bytes. By convention, the maximum value of 2147483647 bytes is interpreted

as 3221225472 bytes (3GB). The default SHMMAX value is 64MB on Redhat 7.1 Linux system. 1 GB is the largest value of each shared memory segment size in bytes. The default value 64MB is enough to small jobs but cause many large NWCHEM jobs crashed. Since RedHat 7.1 needs 100MB memory to run most its processes and there are several applications such as PBS and job accounting needs around 100MB, we decided to change the value of SHMMAX from 64MB to 256MB. This change solved the problem of large jobs on the mixed processor cluster.

Performance comparison using mixed processors and unique processors

Since the network bandwidth plays a major role in the performance, the difference of performance between a mixed processor job and a unique processor job is not very significant when the number of the processors is increasing. We have run some testing jobs with NWCHEM. The results show that the wall-time of a unique processor job is about 20% less than the mixed processor job when the number of processors is less than 4. It reduced to less than 10% when the number of the processors is increased to 16.

Conclusion

In many cases, the simple test-bed cluster will be upgraded to a mixed processor cluster. Most of money is spent on the goal to achieve high computation and communication performance. The configuration of I/O hardware for the machine is insufficient. The faster processors and high-speed communication tend to be more important to many people when they make decision on how to configure the new cluster. It is hard to say how much I/O hardware or how many dedicated I/O nodes are sufficient. In many cases, the application's requirements play an important role on I/O hardware configuration. Sometimes, the system architecture and quality of the I/O hardware might also play role.

The fast file systems can help to improve the mixed processor cluster's performance. There are several fast file systems that are still investigated for the cluster. In many cases, the home directory and application directory are NFS mounted, and can be easily accessed from other machine due to the convenience feature of NFS. But NFS is extremely slow. The read/write data operation of a parallel application on the NFS directory will be bottleneck for the application.

Some applications such as MPI and Linda can have multiple processes access a common file. Our experience has been that, performing I/O from multiple processes on the mixed processor cluster can achieve higher performance.

References:

1. D. E. Bernholdt, E. Apra, H. A. Fruchtl, M.F. Guest, R. J. Harrison, R. A. Kendall, R. A. Kutteh, X. Long, J. B. Nicholas, J. A. Nichols, H. L. Taylor, A. T. Wong, G. I. Fann, R. J. Littlefield and J. Nieplocha, "Parallel Computational Chemistry Made Easier: The Development of NWChem", Int. J. Quantum Chem. Symposium 29, 475-483 (1995).

2. R. J. Harrison, J. A. Nichols, T. P. Straatsma, M. Dupuis, E. J. Bylaska, G. I. Fann, T. L. Windus, E. Apra, J. Anchell, D. Bernholdt, P. Borowski, T. Clark, D. Clerc, H. Dachsel, B. de Jong, M. Deegan, K. Dyall, D. Elwood, H. Fruchtl, E. Glendenning, M. Gutowski, A. Hess, J. Jaffe, B. Johnson, J. Ju, R. Kendall, R. Kobayashi, R, Kutteh, Z. lin, R. Littlefield, X. Long, B. Meng, J. Nieplocha, S. Niu, M. Rosing, G. Sandrone, M. Stave, H. Taylor, G. Thomas, J. van Lenthe, K. Wolinski, A. Wong, and Z. Zhang, "NWChem, A Computational Chemistry Package for Parallel Computers, Version 4.0.1" (2001), Pacific Northwest National Laboratory, Richland, Washington 99352-0999, USA.

3. Peter M. Chen, Edward K. Lee, Garth A. Gibson, Randy H. Katz, and David A. Patterson, "RAID: high-performance, reliable secondary storage", ACM Computing Surverys, 26(2):145-185, June 1994.

4. http://www.parl.clemson.edu/pvfs