Experiences with LSF and cpusets on the Origin3800 at Dresden University of Technology

Stefanie Maletti
TU Dresden, University Computer Center (URZ)
stefanie.maletti@urz.tu-dresden.de

ABSTRACT: Based on the long experience with Cray and SGI machines (Cray T3E and SGI Origin2800) in a complex application-oriented environment and the usage of these machines for parallel applications, an operational concept was designed for the Origin3800 at Dresden University of Technology. The exclusive usage of processors by means of LSF and cpusets is the main feature of this concept. We present our initial problems concerning the installation and configuration of LSF 4.1 with cpusets up to the enforcement of our operational concept.

KEYWORDS: LSF, CPUSET, Origin3800

1. Introduction

Dresden University of Technology (TU Dresden) is one of the largest universities in the eastern part of Germany with more than 28 000 students and about 4500 employees focused in engineering sciences, humanities and social sciences, natural sciences and medicine. In December 1996, the TU Dresden acquired one of the first Origin2000s shipped to Europe. In order to prepare the engineers and scientists of the university in the use of the Origin2000 and future parallel computers, the Center of High Performance Computing (ZHR) was founded in 1997. The ZHR provides support and consultations for analysis and performance optimisation for all users of the high-performance computers.
The University Computer Center (URZ) houses the university's high-performance computers. It is responsible for technical support, installations and maintenance, as well as user and software administration. This is carried out in very close cooperation with ZHR.

2. Overview of the HPC Resources

With the installation of the fourth high-performance computer in January 2002, the TU Dresden closed the HPC project, which was originally initiated in 1994. The two-step project was realized by SGI – the first step was the Origin2800 in December 1996 and the second step started with the Cray T3E in October 1998 as a placeholder for the new Origin3800 system, which was under development at that time. In August 2000, we were proud to receive the first 64-CPU Origin3800 in Europe! With a delay of 4 months, the next 64 CPUs were provided as a separate computer. In March 2001, the two systems were connected together to form a single system image with 128 processors. The next step was planned to be the change of 64 processors to Itanium IA 64. However, because of the delay in Intel's chip delivery of Merced, we decided, together with SGI, to install another 64-CPU MIPS Origin3800 system instead of the Itanium upgrade.

SGI Origin2800 – rapunzel

- 48 CPUs MIPS R10000 with 195 MHz, 4 MB secondary cache
- 17 GB memory
- 370 GB storage
- IRIX 6.5.14
- NQE 3.3.0.15

SGI Onyx2 – rapinzel

- 8 CPUs MIPS R10000

Cray T3E - ratbert

- 64 PEs (+ 2 Com, 2 OS)
- 128 MB memory per PE
- 172 GB storage
- UNICOS/mk 2.0.5
- NQE 3.3.0.13

Early SGI Origin3800 System

- 64 CPUs MIPS R12000 with 400 MHz, 8 MB secondary cache
- 32 GB Memory
- Serial number L2000001 – first in Europe

SGI Origin3800 Today

128-CPU system romulus

- 128 CPUs MIPS R12000 with 400 MHz, 8 MB Secondary Cache
- 64 GB memory
- 500 GB storage

64-CPU system remus

- 64 CPUs MIPS R12000 with 400 MHz, 8 MB Secondary Cache
- 64 GB memory
- 300 GB storage

Software:

- IRIX 6.5.14f
- LSF 4.2

The Origin2800 is still used as a compute server for general purpose applications. Approximately 500 users from the whole campus can submit jobs to this resource. The applications, covering technical fields such as FEM, CFD, chemistry and mathematics, usually utilize the following software, but also self-written programs:

- **FEM:** ABAQUS, ANSYS, MARC, NASTRAN
- **CFD :** CFX, Fluent, Tascflow
- **Chemistry:** GAUSSIAN, GAMESS, MOLPRO
- **Math. Libraries:** NAG, Maple, MATLAB, Mathematica.

Although the development of parallel programs for the Origin system was encouraged by the Center of High Performance Computing, our customers were not pleased with the performance of multithreaded jobs if they had to compete with all other sequential jobs. We observed - like many other Origin sites at that time - that the turnaround time of parallel jobs (both shared memory and MPI applications), significantly varied from the time needed on a dedicated system.
For time critical jobs, the batch scheduler miser, providing resources and controlling the optimal execution had been a way out. However, sometimes miser failed to get even those resources and then the throughput was critical again.
The conclusions of these observations became the prerequisites for the purchase of the second step of our HPC project, and finally they were found in our operational concept for the Origin3800.

The Cray T3E - one of the fastest computer systems in the past - with the supercomputer operating system UNICOS/mk and the batch system NQE has been a very successful example for development and production environment for parallel jobs. Therefore, the T3E closed the gap the Origin2800 had left. Today, the T3E is still used as a parallel server for selected projects, but with regressing usage in dependency with the growing usage of the Origin3800.
Nevertheless, our wishes for the successor of the Origin2000 are compared to the proven performance of the T3E.

At present, we run two Origin3800 systems: the larger system with 128 processors as a parallel server and the 64-CPU system for large memory applications using nearly the same operational concept.

3. Operational Concept of the Origin3800

In cooperation with ZHR, we designed the following operational concept for the 128-CPU system romulus:

- parallel computer for projects;
- exclusive usage of processors and memory using LSF and cpusets;
- boot cpuset using 8 processors for the operating system and interactive tasks;
- different day and night/weekend operational mode by means of the checkpoint/restart facility:
    - Express mode supporting application development and small production jobs during daytime from 7:00 a.m. to 8:00 p.m.
    - Large production jobs during the night and at the weekend.

Queue details are specified as follows:

- **Express queue:** application development and small production jobs with a maximum CPU time of 10 minutes and up to 16 processors.
- **Short queue:** production jobs with a maximum CPU time of 5 hours and up to 64 processors per user.
- **Long queue:** production jobs with a maximum CPU time of 5 hours and up to 120 processors per user.

The goal of the concept described is to guarantee a reliable batch operation with support of multiple parallel jobs from several users without interference of personnel. The turnaround time of parallel jobs should not be allowed to exceed the time on a dedicated system by more than 10 percent.

The exclusive usage of the processors and their local memory is the main feature of the concept. The IRIX cpuset feature embedded in LSF 4.1 provides the possibility to enforce the concept because cpusets guarantee the resources better than miser could.

4. Learning about LSF with cpusets

Our experience with the batch system LSF started with version 3.2, which was shipped with the first 64-CPU Origin3800 system in August 2000 and led to good impressions in comparison to NQE, which we have been using on Cray T3E for years. During the acceptance period of the second 64-CPU Origin3800 system, the configuration of LSF 4.1, which was installed in January 2001 in connection with cpusets, was a major task and at that time very difficult. Initial installation and configuration of LSF cpuset integration could not be carried out without special knowledge obtained from SGI and Platform Computing personnel. Approximately 6 months elapsed before we were able to establish stable user operation. In the same time, the two systems were also connected together and experienced some hardware failures.
From the operational point of view we observed the following aspects:

1. Because cpusets were not assigned exclusively, threads left their cpuset.
2. Memory was not assigned exclusively.
3. MPI jobs running with the MPI_DSM_MUSTRUN environment variable set, returned errors.
4. A mixture of both sequential and parallel jobs led to a fragmentation of the system after some time (about one week), i.e. contiguous cpusets were no longer possible.

Analysis of the issues yielded three problem categories:

1. LSF configuration.
2. Memory.
3. A mixture of sequential and parallel jobs.

As a result of the combination of two 64-CPU systems with different memory capacity, we temporarily obtained a 128-CPU system with unequally distributed memory. This was just another problem of the second category for the batch system LSF, because LSF handles all CPUs in the same manner. We took care of this problem until the delivery of the remaining memory.

The configuration of LSF needed support again from SGI. Checking the setting of the cpuset options, we did not find exclusively assigned cpusets until the cpuset options were configured on the command line. Writing down the options in the configuration file lsf.conf as usual would not work!
At first we chose the cpuset options

- CPUSET_CPU_EXCLUSIVE
- CPUSET_MEMORY_LOCAL
- CPUSET_MEMORY_EXCLUSIVE

Entering these options with the command line as follows
bsub –exsched "CPUSET_OPTIONS=CPUSET_CPU_EXCLUSIVE|
CPUSET_MEMORY_LOCAL| CPUSET_MEMORY_EXCLUSIVE" …
we solved the first problem and found no threads outside of their cpuset. However, the memory was still not exclusive.
The described memory options caused the following behaviour:

- Job A requires more memory than its assigned cpuset provides;
- Job A is assigned vacant CPUs;
- Job B is assigned a cpuset without memory because LSF does not check the memory of free processors;
- Job B (an MPI job) fails, as it is fixed to its cpuset by MPI_DSM_MUSTRUN.

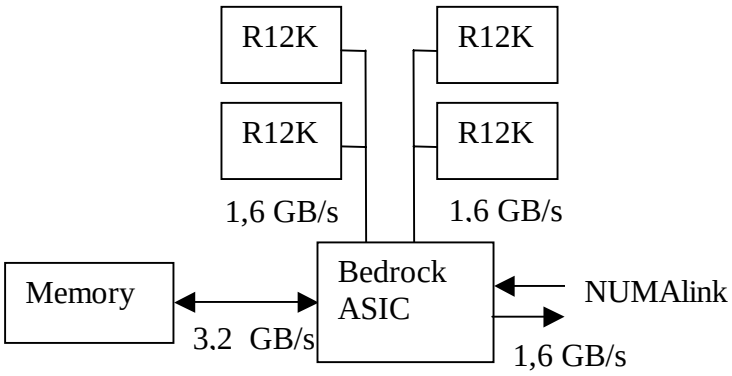
To solve this problem, so that the successor job will fail instead of the originator, we now use the new cpuset options:

- CPUSET_CPU_EXCLUSIVE
- CPUSET_MEMORY_LOCAL
- CPUSET_MEMORY_MANDATORY
- CPUSET_POLICY_KILL

These options guarantee that only memory of nodes within the cpuset can be allocated and the process will be killed if more memory is needed but not available within the cpuset.
As a consequence, the user has to request as many processors as memory is required. In other words, the user has to "pay" for the memory in the same manner as he "pays" for the CPUs. As a further consequence, all nodes should have homogeneous memory capacity to work with LSF and cpusets.

5. On the Way to the Enforcement of Our Operational Concept

As we learned by configuration of LSF, the main feature of our operational concept – the exclusive usage of processors and memory - works in LSF 4.1. However, the memory still remains a difficulty!
For the Origin3800, the memory is physically distributed onto nodes or so-called C bricks. Four CPUs share the same memory block. From the memory point of view, this is the smallest unit. However, a cpuset can also be built with one CPU and this, therefore leads to bottle necks for sequential jobs with large memory allocation or jobs with a number of processors that isnot a multiple of 4. For example, jobs with 1, 2 or 3 processors can assign the memory from the entire C brick. The policy "kill" does not work until the memory of the entire C brick is exhausted.



Another problem can occur with interactive or system tasks which are bound to the boot cpuset. The boot cpuset is not exclusive. Memory allocation can be carried out from outside the boot cpuset. We observed a situation where the policy “kill” (no more memory within the cpuset) was applied to a job that was proven to run with the assigned cpuset. The conclusion led to an interactive task with large memory allocation.

In the next step, we want to describe the phenomena of fragmentation. If a mixture of sequential and parallel jobs is running over a certain time (we observed a week), the situation can occur that LSF can not assign contiguous cpusets and some jobs will fail.
On the T3E this will automatically be repaired by job migration. The workaround today on the Origin3800 is carried out with checkpoint/restart. Therefore, the interaction of personnel is required, which must checkpoint as many sequential jobs as necessary to free contiguous cpusets for parallel jobs. Until the parallel jobs are running, the sequential jobs can restart in new cpusets.

Studying LSF, the checkpoint/restart facility plays a very important role for our operational concept, especially if you keep in mind the operational modes we described above. Realising the switch between day and night/weekend mode, we investigated queue windows, suspend and checkpoint/restart.

1. Queue window:
   - The system runs empty, because a new job is not started with respect to the expected run limit exceeding its time frame.
   - Setting the “IGNORE_DEADLINE” parameter, jobs are suspended when the time frame ends.
2. Suspend:
   - The respective resources (cpusets in LSF 4.1, memory in general) are not released.
   - This is contrary to our operational concept and thus not acceptable in general.
   - Nevertheless, suspend will be applied for LSF maintenance.
3. Checkpoint/restart:
   - The only way to switch between queues!
   - For administration purposes, checkpoint/restart is substituted by the automatic migration facility.

During the acceptance period of the second 64-CPU Origin3800 system, we asked for the releasing of cpusets by means of suspend and their reattach at a later time. The answer from SGI was the cpuset reattach feature (cpusetAttachPID, cpusetDeattachPID) in IRIX 6.5.13. In the current version 4.2 of LSF, Platform Computing also supports this feature.

Our switch between day and night/weekend mode is realised by switching the queue layout with a time-controlled script triggering three activities:

1. De-activate all queues;
2. Migrate jobs which are still running and re-queue jobs;
3. Activate the required queues.

Two substantial prerequisites are required for this procedure: Checkpoints have to be allowed for the respective queues, which can be carried out by the administrator, and checkpointing has to be enabled, especially for MPI jobs. The latter is normally carried out at user level. We use a modified mpirun script for this purpose, which also sets the environment variable MPI_DSM_MUSTRUN.
However, the procedure fails as a whole if just one job can not be checkpointed, for whatever reason and this still happens to us. We have at least one application that can not be checkpointed. Fortunately, the current system usage does not require a real day/night mode switch, but a solution is certainly needed and we are still investigating.

Final remarks to checkpoint/restart: Working fine with LSF 4.1 (only problems with IRIX 6.5.11). We are looking forward to the early release of LSF 4.2 with the embedded cpuset reattach feature. However, we are disappointed that while the cpuset reattach feature is working fine now, checkpoint/restart fails. Checkpoints are written, but LSF can not restart jobs because the assigning of cpusets fails.

6. Conclusion

Considering our experiences with the installation, configuration and operation of the Origin3800 we conclude:

1. Parallel applications under the control of LSF and exclusive cpusets get the best possible performance, which is almost the performance of a dedicated system.
2. Switch between day and night/weekend operational mode is not currently realised because one production job can not be checkpointed, but features such as checkpoint/restart and suspend are of great importance and need further assistance.
3. Memory still remains a problem for LSF (inhomogeneous memory capacity; hardware changes need new configuration).
4. Fragmentation problems should be solved automatically.

The resulting wish list includes

- complete resource control with LSF,
- cpuset management under the control of IRIX (one possibility could be the resource pool manager daemon generally provided with IRIX).

The Origin3800 system is a well-balanced distributed shared-memory computer system, which works well with its software and is fully accepted by our end-users. Features such as cpusets show the right way to improve the performance of parallel jobs. Nevertheless, the scheduler and the resource management could be enhanced to provide an improved environment for parallel programming.
The usage of our Origin3800 will continue to grow. At present, we run a workload of 70 %, although we do not accept any oversubscribing of the machine by the use of our restrictive operational concept.