Performance comparison between a massive SMP machine and clusters

Martin Scarcia, Stefano Alberto Russo
Sissa/eLab
joint Democritos/Sissa Laboratory for e-science
Via Beirut 2/4 34151 Trieste, Italy

Stefano Cozzini
CNR/INFN Democritos National Simulation Center and Sissa/eLab
Via Beirut 2/4 34151 Trieste, Italy

July 2009

1 Introduction

In this brief report we compared the performance of selected scientific applications on sun-test a massive SMP 32-core machine with blade and zebra, the two main partitions of the production cluster at SISSA. In tables 1 and 2 some details on the hardware are presented. Information on the interconnection network is available in table 3 along with some network performance results, obtained with the Intel MPI Benchmark.

Our main aim was to assess if a single massive multicore machine is a viable alternative to Beowulf-type clusters when running scientific applications. No considerations regarding hardware costs and power consumption were made. Two identical sun-test machines were also connected via high speed Infiniband network to see if the code scalability was good among two such nodes.

<table>
<thead>
<tr>
<th>name</th>
<th>AMD CPU type</th>
<th>cores/chip</th>
<th>cores/node</th>
</tr>
</thead>
<tbody>
<tr>
<td>zebra</td>
<td>Xeon E5420 2.5GHz</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>blade</td>
<td>Opteron 280 2.4GHz</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>sun-test</td>
<td>Opteron 8384 2.7GHz</td>
<td>4</td>
<td>32</td>
</tr>
</tbody>
</table>

Table 1: Machine details
Table 2: Memory/cache details (L1 per core, L2 per chip, L3 per chip)

<table>
<thead>
<tr>
<th>Name</th>
<th>L1 (KB)</th>
<th>L2 (MB)</th>
<th>L3 (MB)</th>
<th>Type</th>
<th>Speed (MHz)</th>
<th>Total (Gb)</th>
</tr>
</thead>
<tbody>
<tr>
<td>zebra</td>
<td>64</td>
<td>12</td>
<td>0</td>
<td>FB-DIMM</td>
<td>667</td>
<td>16</td>
</tr>
<tr>
<td>blade</td>
<td>128</td>
<td>1</td>
<td>0</td>
<td>DDR</td>
<td>400</td>
<td>8</td>
</tr>
<tr>
<td>sun-test</td>
<td>64</td>
<td>2</td>
<td>6</td>
<td>DDR2</td>
<td>667</td>
<td>64</td>
</tr>
</tbody>
</table>

Table 3: Intel MPI Benchmark results

<table>
<thead>
<tr>
<th>Name</th>
<th>Network</th>
<th>Latency</th>
<th>Peak bw</th>
<th>bw/core</th>
</tr>
</thead>
<tbody>
<tr>
<td>zebra</td>
<td>Infiniband 20G</td>
<td>4</td>
<td>907</td>
<td>113</td>
</tr>
<tr>
<td>blade</td>
<td>Infiniband 10G</td>
<td>5</td>
<td>560</td>
<td>140</td>
</tr>
<tr>
<td>sun-test</td>
<td>Infiniband 10G</td>
<td>n.a.</td>
<td>690</td>
<td>22</td>
</tr>
</tbody>
</table>

2 Results

All the tests performed are part of the eLab Benchmarking Suite and details on the benchmark datasets are provided on the project webpage.

Every test presented in the following sections was run six times and the wall clock time was measured via the `/usr/bin/time` system tool. The figures presented here are all based on average times computed from the timings measured in the different runs.

The tests were always run exclusively on the machine, that is ensuring that no other CPU consuming process was active.

The software was compiled with the same software stack to ensure software uniformity among different machines was respected. In detail the Intel Fortran Compiler 10.1.018 was used in combination with OpenMPI 1.3 libraries.

The first benchmark is based on RegCM, a climate modeling application. In figure 2 are displayed the average wall-clock times expressed in seconds. What we see is that sun-test provides the best absolute best performance when using between 8 and 16 CPUs, when using more it lags behind both zebra and blade. In fact as it can be seen in figure 2 the scalability drops beyond 16 CPUs.
Figure 1: RegCM results
Figure 2: *RegCM* scalability
The second benchmark is based on a diffusion quantum Monte-Carlo code for super fluids (QMCSF), which is known to scale almost linearly provided the test case is big enough. In figure 3 we see that sun-test performs better than blade, but worse than zebra. Scalability is almost optimal on all three machines and there is no noticeable difference among the three (figure 4).

![QMCSF results overview](image)

Figure 3: QMCSF results overview

The third benchmark is based on Gromacs a classical molecular dynamics code widely used for biophysics studies. The code is known to have good scalability when using a relatively small number (tens) of CPUs. If we take a look at figure 5 we see that sun-test provides the absolute best results in the 8-32 CPU range, but becomes the worst performer when using 64 CPUs. This is evident also in figure 6 where we see a clear drop in scalability between 32 and 64 CPUs. This lack of performance is clearly due to the insufficient per process network bandwidth (see table 3) and could be overcome using a faster network connection.

The last benchmark is based on NAMD, another classical molecular dynamics application. Its performance is very similar to that of Gromacs as
Figure 4: *QMCSF* scalability
Figure 5: Gromacs results overview
Figure 6: *Gromacs* scalability
can be seen in figure 7 and 8. The three systems perform are comparable up to 32 CPUs, where *sun-test* scalability drops due to the insufficient network bandwidth.

![Figure 7: NAMD scalability](image)

### 3 Conclusions

Though it might be tempting to substitute small clusters in the tens of CPUs range with massive multicore SMP systems this report shows that machines such as *sun-test* are not always an equally performing substitute.

Results show that codes, which rely heavily on interprocess communication as *RegCM* are unable to exploit fully all of the cores likely due to memory access conflicts and perform definitely better on a cluster type architecture.

Less communication-bound codes as quantum Monte Carlo or classical MD are likely to perform equally well on both SMP machines and clusters. In any case it is important to note that when connecting two multicore machines with a relatively big number of cores, the network interconnect
Figure 8: NAMD performance overview
is rapidly saturated from the point of view of a single process. Therefore a network that performs well in a cluster might not be up to the task of connecting two massive SMP machines.