ScaleMP Applications

at RWTH Aachen University

Christian Terboven, Dirk Schmidl, Dieter an Mey
{terboven, schmidl, anmey}@rz.rwth-aachen.de

a HPC case study with Dell 01.06.2010, ISC 2010
Hamburg, Germany
Agenda

- The ScaleMP vSMP Architecture
- System Examination
  - vSMP Utilities
  - Synthetic Benchmarks
  - Kernels
- Applications on ScaleMP
  - FIRE
  - SHEMAT-Suite
  - TrajSearch
- Conclusion
The ScaleMP vSMP Architecture
The ScaleMP system at RWTH Aachen

- In total 104 cores, 170 GB of memory, 1.4 TB of disc space

InfiniBand Switch

Intel Xeon E5420 @ 2.5 GHz

Board 0

13 Boards

16 GB RAM

80 GB HDD

80 GB HDD
vSMP: Cache-coherent aggregation of physical resources via the InfiniBand network (employing virtualization techniques)

- The OS view (Single System Image):
  - 170 GB RAM
  - 1.4 TB Raid Device

- Shared-Memory parallelization on a cluster of x86-based boards
  - Cheaper and more flexible than hardware-based solutions
  - Strong cc-NUMA characteristics
System Examination
### vSMP utilities: vsmpstat

$ vsmpstat –bbc

[...]  

Board Basic Counters:

<table>
<thead>
<tr>
<th>BBC</th>
<th>Time since reset</th>
<th>%vSMP</th>
<th>TLB Flush</th>
<th>PT Write</th>
<th>PT WrtEm</th>
<th>HPET Resync</th>
<th>Drift (uS)</th>
<th>Frames %Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>bbc:00:00</td>
<td>899802</td>
<td>13.1</td>
<td>3440358</td>
<td>480597</td>
<td>255081</td>
<td>0.000</td>
<td>0.000</td>
<td>4005935</td>
</tr>
<tr>
<td>bbc:00:01</td>
<td>899801</td>
<td>4.0</td>
<td>301449</td>
<td>435875</td>
<td>330412</td>
<td>0.000</td>
<td>0.000</td>
<td>4005935</td>
</tr>
<tr>
<td>bbc:00:02</td>
<td>899803</td>
<td>4.3</td>
<td>505126</td>
<td>401652</td>
<td>211529</td>
<td>0.000</td>
<td>0.000</td>
<td>4005935</td>
</tr>
<tr>
<td>bbc:00:03</td>
<td>899805</td>
<td>4.5</td>
<td>608419</td>
<td>726302</td>
<td>485302</td>
<td>0.000</td>
<td>0.000</td>
<td>4005935</td>
</tr>
</tbody>
</table>

[...]

- Innovative architectures require tools to examine them properly
  - ScaleMP: Memory traffic between boards
  - ScaleMP: Process and thread migration + memory usage
Distributed-Shared-Memory systems can be characterized by their „numaness“ (ration between local and remote memory access time):

- **read_from_other**: Time to read a page that was written by another thread, both threads are on separate boards
- **write_from_other**: Time to write to a page that was written by another thread before, both threads are on separate boards
- **write_self**: Time to write to self-written page (test for vSMP overhead)

<table>
<thead>
<tr>
<th></th>
<th>ScaleMP system in AC</th>
<th>4s Intel Tigerton machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>read_from_other</td>
<td>43.37</td>
<td>1.76</td>
</tr>
<tr>
<td>write_from_other</td>
<td>40.44</td>
<td>2.29</td>
</tr>
<tr>
<td>write_self</td>
<td>2.34</td>
<td>2.10</td>
</tr>
</tbody>
</table>

Page Access Benchmark results in microseconds [usec], two threads
OpenMP-parallelized version of STREAM w/ first-touch initialization

- Red line: 104 threads, scheduling left to the system
- Blue line: 104 threads, explicit thread binding

400 iterations, time has been measured for each one individually.

ScaleMP scheduler is improving over time, but does not reach optimum.

Theoretical peak: 8.5 GB/s per boards => 110.5 GB/s
Applications on ScaleMP
Image Retrieval: A set of query images is compared with all images in a (huge) database and the k most similar images are returned

- Performance comparison of common features on different databases
- Analysis of correlation of different features
- Nested Parallelization – Outer level: Queries, Inner level: Comparison

Data Mining is well-suited for Shared-Memory parallelization, but hard with MPI

Thomas Deselaers and Daniel Keysers,
RWTH I6: Chair for Human Language Technology and Pattern Recognition
Nested Parallelization improves efficiency by reducing the total overhead.

Explicit Thread Binding: `export KMP_AFFINITY=scatter`
SHEMAT-Suite

- Geothermal Simulation Package to simulate groundwater flow, heat transport, and the transport of reactive solutes in porous media at high temperatures (3D)
  - Forward simulation
    - 3D finite-differences solver, Coupled transient equations for groundwater flow, Compute state variables from rock properties
  - Inverse computation
    - Parameter estimation
- Written in Fortran, two levels of parallelism
  - Independent Computations of the Directional Derivatives
  - Setup and Solving of linear equation systems
vSMP reduces memory allocation performance, therefore the code has been modified to re-use arrays instead of re-allocation. Scalability is near the theoretical peak. Simulation time has been reduced from one month to less than a day! No MPI involved 😊
TrajSearch: Analysis of Turbulent Flow Fields

- Analysis of DNS (Direct Numerical Simulation) output generated on IBM BlueGene for a $2048^3$ grid. $\frac{1}{2}$ year with 16K MPI processes.
- Analysis of geometric properties at micro-scale with the Gradient Trajectory Method by grouping trajectories starting at each grid point which share the same minimum and maximum.
- Not suited for MPI because of extreme load imbalances. Thus parallelized with OpenMP for large Shared Memory Machines.
- Parallelization on ScaleMP with up $512^3$ grid points leads to 80X speed-up on 104 cores on the ScaleMP machine.
Conclusion
Summary and Conclusion

- For some applications Shared-Memory is just the right parallelization paradigm and ScaleMP offers a nice solution
  - Applications might need multiple levels of parallelization

- ScaleMP has proven to deliver good performance
  - Shared-Memory machine with strong cc-NUMA characteristics
  - Shared-Memory in software is cheaper than in hardware (but more cc-NUMA)

- What’s still missing and subject to further research:
  - Experience with Tuning Methodologies on ScaleMP
    - Does manual data “prefetching“ help?
    - Sometimes the optimal thread binding is “surprising“: lack of binding support for nested OpenMP in the Intel compiler.
Thank you for your attention.