Evaluation

Georg Hager, Johannes Habich (RRZE)
Stefan Donath (Lehrstuhl für Systemsimulation)

Universität Erlangen-Nürnberg

ZKI AK Supercomputing 25./26.10.2007, GWDG
Outline

- Administrative issues
  - Installing the Head Node
  - Cluster Network Topology
  - RIS-Unattended Installation
  - Domain integration
  - User Environment
  - Benchmarks for Intel MPI PingPong

- Current user projects
  - Evaluating Excel integration
    - LINPACK sample
    - Homebrew VBA macros for simple Jacobi benchmark
  - NUMA and affinity issues

- Conclusions
Opteron Test Platform

- 7 quad Opteron nodes (Dual Core Dual Socket)
- 4 GB per node
- 8GB on head node
- Windows 2003 Enterprise + Compute Cluster Pack
- Visual Studio 2005, Intel compilers, MKL, ACML
- Star-CD
- Gbit Ethernet

- Access via RDP or ssh (sshd from Cygwin)
  - GUI tool for job control: Cluster Job Manager
  - CLI: job.cmd script
Cluster Layout

- **Plan**: `ccsfront` as VMWare virtual machine on master node
  - Only machine accessible to users
  - Easily movable

- **However**: Serious problems with Visual Studio and Intel compilers inside VM
  - Same procedures work on `ccsmaster`
  - Being investigated

- For now, people work on `ccsmaster`
Installing the Head Node

- Preinstalled headnode and cluster nodes from transtec with evaluation version of WinCCS2003
- Preliminary network connection bandwidth evaluation with Intel IMB benchmarks
- **No SUA support → Clean installation of Win2003 Enterprise R2 x64**

- Installation of Intel Fortran and C compiler with Visual Studio 2005 integration
  - Intel C++ Compiler 9.1 and 10.0
  - Intel Fortran Compiler 9.1
  - Intel MKL 9.0
Cluster Network Topology

- Separate private and public 1GE networks available
- DHCP Server could not separate the two scopes to two physical network adapters
- DHCP Server is reconfigured without warning for Remote Installation Services (RIS) to install nodes unattended

→ only private network was used, with NAT translation for outside communication
RIS Unattended Installation

- Creating the RIS (Remote Installation Server) Image from Win2003 installation CD
- By-hand inlining of R2 necessary packets and settings from Win2003 installation CD 2
- Adjustment of wrong paths inside the configuration files
- First attempt to install RIS caused a complete DHCP breakdown, as RIS changes complete DHCP configuration and launches without proper rights for standard scope 192.0.0.x
- After that, RIS installed compute nodes flawlessly
RIS Unattended Installation

- Domain Administrator rights are necessary for creating automatically new computer accounts inside domain
- RIS deploy wizard warns user that **password is visible in plain text** during deployment → huge security risk during each setup!

**Suggestions:**
- No plain text passwords
- Easy wizard creation of standard Win2003 images
- DHCP Server with multiple instances for each network interface
- Failure messages should be more elaborate
Domain Integration

- Headnode and cluster nodes were integrated into UNI-Erlangen ADS
- Problem of supplying a working directory which is both available as a network share via Windows Explorer and available to the jobs running on the cluster
  - User works on mounted network share
  - Job works on complete UNC path leading to same network share
  - One common path for both, but UNC for users is not a preferred choice
  - Automatic mapping of this location at job start and user login
  - Problems of some Windows software to work with UNC (CMD.exe, partly Visual Studio 2005)
Intel MPI Ping Pong benchmark

- Compiling inside SUA and Visual Studio 2005

- Issuing jobs with standard MPIEXEC causes 4 threads to run on each node → internode communication measured

- Hence: “Fun and interesting ways to run MPI Jobs on CCS” from http://windowshpc.net/

Now:
pernode.bat script hacks %CCP_NODES% system variable and only one task is executed per compute node

→ Measuring real internode connection bandwidth and latency
→ Desirable: flexible specification for „processors per node“ at job submission and runtime
Intel MPI Ping Pong benchmark

PingPong
INTEL MPI Benchmark on CCS
(small packets / latency test)

MS: Variations due to “interrupt coalescing” in GE driver
→ No option to switch off in our driver 😞
Projects on the WinCCS Cluster

- Crack propagation (FAU, Materials Science)
  - F90/OpenMP, C++ (OpenMP to come)

- VirtualFluids (TU Braunschweig)
  - C++/MPI

- Star-CD (FAU, Fluid Mechanics)

- waLBerla (FAU, System Simulation)
  - C++/MPI
**Widely applicable lattice Boltzmann from Erlangen**

- CFD project based on lattice Boltzmann method
- Modular software concept
  - Supports various applications, currently planned:
    - Blood flow in aneurysms
    - Moving particles and agglomerates
    - Free surfaces to simulate foams, fuel cells, a.m.m.
    - Charged colloids
    - Arbitrary combinations of above
- Integration in efficient massive-parallel environment
- Standardized input and output routines
- User-friendly interface
- Platform independency with CMAKE
Widely applicable lattice Boltzmann from Erlangen

- **Porting issues concerning CMake:**
  - CMake has to be configured to find MPI
  - Not possible to specify Cluster Debugger Configurations via CMake (overwrites settings when project is built)

- **Visual Studio & Queues**
  - Not possible to automatically submit and debug parallel job via Visual Studio

- **Debugging issues**
  - MPI Cluster Debugger: Configuration pain to run jobs on remote sites
  - Remote Debugger not able to connect to queued jobs
Goals

- **WCCS as a development environment**
  - Visual Studio
  - Parallel debugging
  - Different compilers
  - Some issues (Intel project system, parallel debugging), but ok in general

- **Coupling of CCS cluster to MS Excel by use of VBA**
  - Job construct, submit
  - Result retrieval
  - Visualization

- **Behaviour of Windows on a ccNUMA architecture**
  - Locality & affinity issues
  - Buffer cache
Excel-CCS coupling

- **CCS API** available for C++ and VBA
  - Documentation issues for VBA, but many examples online:
    
  
- API is part of *Office Professional* only

- Example Excel Sheet and VBA macros for LINPACK parameter scans provided

- **Toy project**: Make Excel sheet for simple heat equation solver with graphical output of performance numbers
Evaluating Excel Integration (LINPACK Example)

- Taking precompiled binaries offered by MS with ACML
- Tuning LINPACK parameters as suggested in: “Hands-On Lab – Building HPC LINPACK Tool”
- Issuing jobs directly to Job Manager
- Querying jobs
- Results are instantly plotted in Excel
Excel-CCS coupling

- **Principles of operation**
  - Provide *Excel worksheet* with necessary parameters
    - Binary name, working dir
    - Number of CPUs, walltime limit
    - Input parameters for application
  - Position *active elements* (buttons,…) linked to VBA macros
  - VBA communicates with *CCS* using *XML*
    - First time generation of XML structure from
      - XML Schema
      - template file (saved from Job Manager app.)
      - scratch
    - Link entries to worksheet cells
  - Use *VBA-CCS API* to construct and submit jobs
    - Many options possible
  - Collect and parse output data and fill cells
    - Simple visualization possible using *Excel graphs*
job.xml template

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Job xmlns:ns1="http://www.microsoft.com/ComputeCluster/"
    Name="job1"
    MaximumNumberOfProcessors="4"
    MinimumNumberOfProcessors="4"
    Owner="unrz55" Priority="Normal" Project="Jacobi"
    Runtime="Infinite">
    <ns1:Tasks>
        <ns1:Task Name="task1"
            CommandLine="echo 10 10 5 | heat_ccs.exe"
            MaximumNumberOfProcessors="4"
            MinimumNumberOfProcessors="4"
            Runtime="Infinite"
            WorkDirectory="\\Ccsmaster\ccsshare\unrz55\x1sttest"
            Stderr="3700-3700-50-err.out"
            Stdout="3700-3700-50-heat.out" />
    </ns1:Tasks>
</Job>
### Cluster Parameters

<table>
<thead>
<tr>
<th>command</th>
<th>heal_ccs.exe</th>
</tr>
</thead>
<tbody>
<tr>
<td>job</td>
<td>job1</td>
</tr>
<tr>
<td>id</td>
<td></td>
</tr>
<tr>
<td>status</td>
<td></td>
</tr>
<tr>
<td>owner</td>
<td>unrz55</td>
</tr>
<tr>
<td>project</td>
<td></td>
</tr>
<tr>
<td>runtime</td>
<td>Infinite</td>
</tr>
<tr>
<td>fill cancel</td>
<td></td>
</tr>
<tr>
<td>backfill</td>
<td></td>
</tr>
<tr>
<td>min CPU</td>
<td>4</td>
</tr>
<tr>
<td>max CPU</td>
<td>4</td>
</tr>
<tr>
<td>asked</td>
<td>4</td>
</tr>
<tr>
<td>allocated</td>
<td>exclusive</td>
</tr>
<tr>
<td>exclusive</td>
<td></td>
</tr>
<tr>
<td>priority</td>
<td>normal reboot</td>
</tr>
<tr>
<td>license</td>
<td></td>
</tr>
<tr>
<td>cluster</td>
<td>ccsmaster</td>
</tr>
<tr>
<td>work dir</td>
<td></td>
</tr>
<tr>
<td>input file</td>
<td></td>
</tr>
<tr>
<td>output file</td>
<td>heat.out</td>
</tr>
</tbody>
</table>

### Run
- Successful

### Query

### Queue

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>status</th>
<th>priority</th>
<th>CPUs</th>
</tr>
</thead>
<tbody>
<tr>
<td>380</td>
<td>platzhalter</td>
<td>Running</td>
<td>Normal</td>
<td>8,8</td>
</tr>
</tbody>
</table>
Submitting a Job with VBA in Excel

' connect to cluster, defined by xls cell “Cluster”
Set objCluster =
    CreateObject("Microsoft.ComputeCluster.Cluster")
objCluster.Connect (Range("Cluster").Value)

' Job object from XML description
Set Job = objCluster.CreateJobFromXml(strXML)

' obtain user credentials
Set WshNetwork = CreateObject("WScript.Network")
UserName = WshNetwork.UserDomain & "\" &
    WshNetwork.UserName

' submit job to queue
ID = objCluster.QueueJob((Job), UserName, "", False, 0)

Many other CCS-API and general VBA functions available
- Status query, job cancel etc.
- VBA: Regexp package …
## Result retrieval e.g. by VBScript Regexp Package

### Results

<table>
<thead>
<tr>
<th>NX</th>
<th>NY</th>
<th>#THR</th>
<th>MLUPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>100</td>
<td>4</td>
<td>625.00</td>
</tr>
<tr>
<td>250</td>
<td>250</td>
<td>4</td>
<td>573.39</td>
</tr>
<tr>
<td>400</td>
<td>400</td>
<td>4</td>
<td>540.00</td>
</tr>
<tr>
<td>550</td>
<td>550</td>
<td>4</td>
<td>158.17</td>
</tr>
<tr>
<td>700</td>
<td>700</td>
<td>4</td>
<td>156.88</td>
</tr>
<tr>
<td>850</td>
<td>850</td>
<td>4</td>
<td>159.32</td>
</tr>
<tr>
<td>1000</td>
<td>1000</td>
<td>4</td>
<td>160.00</td>
</tr>
<tr>
<td>1150</td>
<td>1150</td>
<td>4</td>
<td>161.26</td>
</tr>
<tr>
<td>1300</td>
<td>1300</td>
<td>4</td>
<td>159.13</td>
</tr>
<tr>
<td>1450</td>
<td>1450</td>
<td>4</td>
<td>160.25</td>
</tr>
<tr>
<td>1600</td>
<td>1600</td>
<td>4</td>
<td>160.60</td>
</tr>
<tr>
<td>1750</td>
<td>1750</td>
<td>4</td>
<td>160.68</td>
</tr>
<tr>
<td>1900</td>
<td>1900</td>
<td>4</td>
<td>160.44</td>
</tr>
<tr>
<td>2050</td>
<td>2050</td>
<td>4</td>
<td>160.03</td>
</tr>
<tr>
<td>2200</td>
<td>2200</td>
<td>4</td>
<td>159.63</td>
</tr>
<tr>
<td>2350</td>
<td>2350</td>
<td>4</td>
<td>160.63</td>
</tr>
<tr>
<td>2500</td>
<td>2500</td>
<td>4</td>
<td>160.01</td>
</tr>
<tr>
<td>2650</td>
<td>2650</td>
<td>4</td>
<td>160.55</td>
</tr>
<tr>
<td>2800</td>
<td>2800</td>
<td>4</td>
<td>159.80</td>
</tr>
</tbody>
</table>

### Chart

**MLUPs**

- X-axis: N
- Y-axis: MLUPs

---

25.10.2007  hpc@rrze.uni-erlangen.de  WinCCS@RRZE
Windows on ccNUMA

- Locality and congestion problems on ccNUMA
- “First touch” policy for memory pages ensures local placement
  - Watch OpenMP init loops
- Even if placement is correct, make sure it stays that way
  - “pin” threads/processes to initial sockets
  - Issue with OpenMP and MPI
- To make matters worse, FS buffer cache can fill LDs so that apps must use nonlocal memory
  - Use “memory sweeper” before production
- How does all this work on Windows?
NUMA Placement and Pinning with Heat Conduction Solver (Relaxation)

Pinning benefit is only due to better NUMA locality!

additional pinning: +30%

NUMA placement: +60%

4MB L2 limit
Buffer Cache and Page Placement

How to limit FS buffer cache in Windows?

1GB file write in LD0 before benchmark

2GB working set limit
Future plans

- Test of **Ansys CFX 11**
- Test of **StarCD**
- Test of **Allinea DDTLite for Visual Studio**
- **WalbErla (CFD)** development and evaluation of Windows Performance (see Projects)
- Customized **Excel sheets** for standard production applications (see above)
Conclusions

- “Well-known” development environment with HPC add-ons
- Batch system/scheduler is not “enterprise-class”
- Ease of use (develop/compile/debug/job submit)
- Room for improvement with parallel debugging
- Similar ccNUMA issues as with Linux, same remedies
  - Process/thread pinning absolutely essential
- CCS VBA API is extremely fun to play with
  - May be attractive to production-only users
  - Still lacks some coherent documentation
  - Only available with Office Professional