Grid Simulator with Production Scheduling Algorithms

Miroslav Ruda, Hana Rudová

1 Institute of Computer Science
Masaryk University

2 Faculty of Informatics
Masaryk University

Cracow, 2007
Scheduling Algorithms

Algorithms in Grid simulators
- SimGrid, GridSim, GSSIM, Alea
- development and testing of new algorithms
- comparison of algorithms

Algorithms in production systems
- PBSPro, SGE, Maui, Moab
- in simulators: approximated with FIFO (with backfilling)
- hard to reimplement
  - many rules, features, bugs
  - closed source, algorithms not published

Example
- Information about Excludus "Grid Optimizer"
  - uses innovative real-time scheduling algorithm
  - dynamic adaptive scheduling beyond traditional workload managers

www.excludus.com
Scheduling Algorithms

Algorithms in Grid simulators
- SimGrid, GridSim, GSSIM, Alea
- development and testing of new algorithms
- comparison of algorithms

Algorithms in production systems
- PBSPro, SGE, Maui, Moab
- in simulators: approximated with FIFO (with backfilling)
- hard to reimplement
  - many rules, features, bugs
  - closed source, algorithms not published

Example
- Information about Excludus "Grid Optimizer"
  - uses innovative real-time scheduling algorithm
  - dynamic adaptive scheduling beyond traditional workload managers

www.excludus.com
Scheduling Algorithms

Algorithms in Grid simulators
- SimGrid, GridSim, GSSIM, Alea
- development and testing of new algorithms
- comparison of algorithms

Algorithms in production systems
- PBSPro, SGE, Maui, Moab
- in simulators: approximated with FIFO (with backfilling)
- hard to reimplement
  - many rules, features, bugs
  - closed source, algorithms not published

Example
- Information about Excludus "Grid Optimizer"
  - uses innovative real-time scheduling algorithm
  - dynamic adaptive scheduling beyond traditional workload managers
Simulator with production resource management system

Experiments with PBSPro
- different setup of PBS scheduler
  - queues, priorities, backfilling, . . .
- different setup of worker nodes
  - number of nodes per queue, . . .
- modifications of PBS scheduler
- inclusion of virtual machines into PBS

Future: new scheduler
Simulator with Virtual Machines

- **Worker nodes represented by virtual machines**
- **Standard PBS Server and Scheduler**
  - running on dedicated server
- **Standard PBS Mom**
  - running within each virtual machine
- **Sleep jobs**
  - no cpu/memory consumption
Workloads

- Real workloads
- Czech Grid *META Centrum*
- Extracted from PBS accounting
- 2005-2007

- Jobs submitted with the same requirements
  - on worker nodes
  - to the same queues
  - with original owners, ...

- Time reduction
  - configurable reduce factor (600)
  - expected and real wall-clock time
  - job arrival time
Vserver based virtual machines

- One kernel space - very lightweight
- Similar to
  - Linux chroot or BSD jail, with better protection
- Access limits
  - standard: filesystem
  - added: processes, network devices ...
- No hardware emulation, no paravirtualisation
  - no performance penalty
- Copy On Write filesystem
  - one RO root filesystem, with RW overlay filesystem
- System daemons
  - running only once in hosting environment
Experimental Testbed

- **Current workloads (year 2007)**
  - January 4,700 jobs, March 14,000 jobs, Jan-March 70,000 jobs

- **150 Vserver domains**
  - 16 core AMD machine . . . . can use more physical machines
  - represents 300 nodes . . . . . . . . . . . . . . . . . . . . . . . . . can be extended

- **COW filesystem**
  - 300 MB one system installation
  - 12 GB used to represent 150 virtual machines

- **Virtual machine:** only PBS Mom + sshd

- **Submission of all jobs**
  - without any sleep takes less then 10 minutes

- **Reduce factor 600**
  - 1 month -> 1.5 hours, 1 year -> ≤ 1 day
  - reasonably small simulation overhead
Evaluation Criteria

Standard monitoring during simulation run
- number of running/waiting/done jobs
- number of used nodes

Analysis of accounting data
- Weighted Response Time (WRT)
- Weighted SlowDown Time (WSD)
- Weighted Wait Time (WWT)
- metrics per user, queue
- also structured by number of nodes used by job
Experimental Results for March Workload

Number of running jobs and number of used worker nodes. First simulation with "starvation support", second without.

Number of finished jobs.
Experimental Results for Parallel Jobs I.

**number of running jobs**

![Graph showing number of running jobs over time for different queue configurations.](image)

**number of finished jobs**

![Graph showing number of finished jobs over time for different queue configurations.](image)

**number of finished parallel jobs**

![Graph showing number of finished parallel jobs over time for different queue configurations.](image)
Experimental Results for Parallel Jobs II.

### Wait time based on number of nodes used by job

<table>
<thead>
<tr>
<th>Nodes</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>166</td>
<td>326</td>
<td>261</td>
<td>99</td>
<td>272</td>
</tr>
<tr>
<td>2</td>
<td>40</td>
<td>44</td>
<td>25</td>
<td>244</td>
<td>28</td>
</tr>
<tr>
<td>4</td>
<td>308</td>
<td>337</td>
<td>229</td>
<td>522</td>
<td>352</td>
</tr>
<tr>
<td>8</td>
<td>339</td>
<td>513</td>
<td>252</td>
<td>425</td>
<td>584</td>
</tr>
<tr>
<td>12</td>
<td>434</td>
<td>580</td>
<td>393</td>
<td>615</td>
<td>853</td>
</tr>
<tr>
<td>16</td>
<td>617</td>
<td>343</td>
<td>524</td>
<td>694</td>
<td>355</td>
</tr>
<tr>
<td>28</td>
<td>817</td>
<td>361</td>
<td>758</td>
<td>884</td>
<td>430</td>
</tr>
<tr>
<td>32</td>
<td>820</td>
<td>676</td>
<td>761</td>
<td>857</td>
<td>747</td>
</tr>
<tr>
<td>40</td>
<td>1052</td>
<td>937</td>
<td>1020</td>
<td>808</td>
<td>1048</td>
</tr>
</tbody>
</table>

- **A** one queue
- **B** one queue, starvation support
- **C** two queues
- **D** two queues, strict fifo
- **E** two queues, starvation
Jobs with Preemption

- **Motivation:** better support of parallel jobs or priorities
- Two virtual machines running on physical machine
  - first machine: standard jobs
  - second machine: privileged/parallel jobs
- **Magrathea** allows
  - several VMs running on a single computer
  - jobs submitted directly to VMs
- When job is started in privileged domain, Magrathea
  - suspends job in standard domain (if needed)
  - almost all cpu/memory resources are given to privileged domain (but standard is still running)
- **Support of simulator**
  - Magrathea installed on simulated machines too
  - sleep jobs must respect preemption
Conclusion & Future Work

New Grid simulator

- Inclusion of production resource management system
- New experiments: PBSPro (and other algorithms)
- Novel proposal with virtual machines
- New experiments: scheduling with Magrathea

Future work

- Study: limits of the simulator
  - efficient scheduling algorithms needed
  - cannot use actual load on machines
  - monitoring issues
- New scheduler
#!/bin/bash
reduce=600  #reduce factor

sleep $((($SIMSLEEP/$reduce)))  #gap in workload

sudo $SIMUSER qsub -q $SIMQUEUE
#the same node requirements
-l nodes=$SIMNODESL
-l walltime=$((($SIMREQL/$reduce))) «EOF

#sleep instead of real job
sleep $((($SIMWALL/$reduce)))
EOF
reduce=600
sleep $(($SIMSLEEP/$reduce))
sudo $SIMUSER qsub -q sim$SIMQUEUE
    -l nodes=$SIMNODESL
    -l walltime=$(($SIMREQL/$reduce))«EOF

sleeptime = $(($SIMWALL/$reduce))
while ($sleeptime >0) do
    sleep $sleeptime
    #check long how job has been preempted
    sleeptime=`magrathea-preempted-time`;
done
EOF
Weighted Response Time, SlowDown, and Wait Time

\[
SA_j = \text{reqResources}_j \times (\text{endTime}_j - \text{startTime}_j)
\]

\[
TotalSA = \sum_{j \in \text{Jobs}} SA_j
\]

\[
SD = \frac{(\text{endTime}_j - \text{submitTime}_j)}{\text{runtime}_j}
\]

\[
WRT = \frac{\sum_{j \in \text{Jobs}} (SA_j(\text{endTime}_j - \text{submitTime}_j))}{TotalSA}
\]

\[
WSD = \frac{\sum_{j \in \text{Jobs}} SA_j \times SD_j}{TotalSA}
\]

\[
WWT = \frac{\sum_{j \in \text{Jobs}} SA_j \times (\text{startTime}_j - \text{submitTime}_j)}{TotalSA}
\]