Application-Oriented Scheduling in Multicluster Grids

Ozan Sonmez, Alex Iosup, Nezih Yigitbasi, Dick Epema

Parallel and Distributed Systems Group (PDS)
Department of Software Technology
Faculty EEMCS, Delft, the Netherlands
Problems in grid scheduling (1): system

1. Grid schedulers usually do not own resources themselves
   • They have to negotiate with autonomous local schedulers
   • Authentication/multi-organizational issues

2. Grid schedulers have to interface to different local schedulers
   • Some may have support for reservations, others are queuing-based

3. The set of grid resources is heterogeneous and dynamic
   • Hardware (processor architecture, disk space, network)
   • Basic software (OS, libraries)
   • Grid software (middleware)
   • Resources may fail!!!!!!!
Problems in grid scheduling (2): workloads

4. Workloads are heterogeneous and dynamic:
   • Grid schedulers may not have control over the full workload (multiple submission points)
   • Different application types
   • Jobs may have performance requirements

5. Structure of applications
   • Many different structures (parallel, PSAs, workflows, etc)
The DAS System

- 272 AMD Opteron nodes
  - 792 cores, 1TB memory
- Heterogeneous:
  - 2.2-2.6 GHz
  - single/dual core nodes
- Myrinet-10G (excl. Delft)
- Gigabit Ethernet

TU Delft (68)

Leiden (32)

VU (85 nodes)

UvA/MultimediaN (46)

UvA/VL-e (41)

SURFnet6

10 Gb/s lambdas
The Koala Grid Scheduler

• Developed in the DAS system
• Has been deployed on the DAS-2 in September 2005
• Ported to DAS-3 in April 2007
• Independent from grid middlewares such as Globus
• Runs on top of local schedulers

• **Objectives:**
  • Data and processor co-allocation in grids
  • Supporting different application types
  • Specialized application-oriented scheduling policies

Koala homepage: http://www.st.ewi.tudelft.nl/koala/
Support for Different Application Types & Models

- Parallel Applications
  - MPI, Ibis,...
  - Co-Allocation
  - Malleability

- Parameter Sweep Applications
  - Cycle Scavenging
  - Run as low-priority jobs

- Workflows
Koala’s Architecture
Koala Scheduler

- Enforces **Scheduling Policies**
  - Co-Allocation Policies
    - WF, FCM, CA, CF
  - Malleability Management Policies
    - FPSMA, EGS
  - Cycle Scavenging Policies
    - Equi-All, Equi-PerSite
  - Workflow Management Policies
    - Single-Site, Multiple-Site
Runners

- Extends support for different job types
  - **KRunner**: Globus runner
  - **PRunner**: A simplified job runner
  - **IRunner**: Ibis applications
  - **OMRunner**: OpenMPI applications
  - **MRunner**: Malleable applications (DYNACO framework)
  - **CSRunner**: Parameter-sweep applications
  - **WRunner**: Workflow applications (Directed Acyclic Graphs)
Supprot for co-allocation in Koala

- Simultaneous allocation of resources in multiple clusters
  - Higher system utilizations
  - Lower queue wait times

- Co-allocated applications might be less efficient due to the relatively slow wide-area communications
  - Parallel applications may have different communication characteristics
Co-Allocation Policies

- Policies for **non-fixed jobs**:
  - Worst Fit (**WF**)  
    - Load-aware  
      (balance load in clusters)
  - Close-to-Files (**CF**)  
    - Input-file-location-aware  
      (reduce file-transfer times)
  - Cluster Minimization (**CM**)  
    - Communication-aware  
      (reduce number of wide-area messages)
  - Communication-aware (**CA**)  
    - Communication-aware  
      (decisions based on inter-cluster communication speeds)

Support for Malleable Applications in Koala

- **DYNACO** is a framework for building dynamically adaptable applications; more on [http://dynaco.gforge.inria.fr](http://dynaco.gforge.inria.fr)

- **Malleable Policies:**
  - Favour Previously Started Malleable Applications (FPSMA)
    - grow the earliest started
    - shrink the latest started
  - Equi-Grow & Shrink (EGS)
    - Distribute total amount of growing/shrinking equally among running jobs

Support for PSAs in Koala

Parameter Sweep Application Model

• A single executable that runs for a large set of parameters
  • E.g.; monte-carlo simulations, bioinformatics applications...

• PSAs may run in multiple clusters simultaneously

• We support OGF’s JSDL 1.0 (XML)

Support for PSAs in Koala: Motivation

• How to run *thousands of tasks* in the DAS?

• Issues:
  • 15 min. rule!
  • Observational scheduling
  • System contention

• Run them as **Cycle Scavenging Applications !!**
  • Sets priority classes implicitly
  • No need for personal monitoring
Requirements

1. **Unobtrusiveness**
   Minimal delay for (higher priority) local and grid jobs

2. **Fairness**
   Multiple cycle scavenging applications running concurrently should be assigned comparable CPU-Time

3. **Dynamic Resource Allocation**
   Cycle scavenging applications have to Grow/Shrink at runtime

4. **Efficiency**
   As much use of dynamic resources as possible

5. **Robustness and Fault Tolerance**
   Long-running, complex system: problems will occur, and must be dealt with
System Interaction

**CS Policies:**
- Equi-All: grid-wide basis
- Equi-PerSite: per cluster

**Application Level Scheduling:**
- Pull-based approach
- Shrinkage policy

**Scheduler**
- Registers
- Grow/shrink messages
- Submits

**CS-Runner**
- Submits PSA(s)
- Monitors/informs idle/demanded resources

**Head Node**
- KCM
- Node
- Launcher

**Clusters**
- Deploys, monitors, and preempts tasks

20/10/09
Cycle Scavenging Policy

Equipartition-PerSite (per cluster)

Clusters

C1 (12)  C2 (12)  C3 (24)

CS User-1
CS User-2
CS User-3

20/10/09
Support for Workflows in Koala

- We support **Pegasus-Chimera** job description language

- Single-Site Policy (static decision-taking):
  - Map a workflow to the least loaded cluster

- Multiple-Site Policy (dynamic decision-taking):
  - Submit as many tasks to the
    - least loaded cluster
    - faster cluster
What about prediction-based scheduling?

- **Grids**
  - Multi-site and heterogeneous resource structure
  - Dynamic and heterogeneous workloads
    - Highly variable **job runtimes** and **queue wait times** limit the efficient use of the resources by users

## Grid Workload Traces*

<table>
<thead>
<tr>
<th>Traces</th>
<th>Type</th>
<th># CPUs</th>
<th>Duration (Months)</th>
<th># Tasks</th>
<th>Parallel Jobs</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAS2</td>
<td>Research</td>
<td>400</td>
<td>18</td>
<td>1.1 M</td>
<td>66%</td>
</tr>
<tr>
<td>GRID5000</td>
<td>Research</td>
<td>2500</td>
<td>27</td>
<td>1.0 M</td>
<td>45%</td>
</tr>
<tr>
<td>DAS3</td>
<td>Research</td>
<td>544</td>
<td>18</td>
<td>2 M</td>
<td>15%</td>
</tr>
<tr>
<td>SHARCNET</td>
<td>Research</td>
<td>6828</td>
<td>12</td>
<td>1.2 M</td>
<td>10%</td>
</tr>
<tr>
<td>AUVER</td>
<td>Production</td>
<td>475</td>
<td>12</td>
<td>0.4 M</td>
<td>0%</td>
</tr>
<tr>
<td>NORDU</td>
<td>Production</td>
<td>2000</td>
<td>24</td>
<td>0.8 M</td>
<td>0%</td>
</tr>
<tr>
<td>LCG</td>
<td>Production</td>
<td>24515</td>
<td>4</td>
<td>0.2 M</td>
<td>0%</td>
</tr>
<tr>
<td>NGS</td>
<td>Production</td>
<td>-</td>
<td>6</td>
<td>0.6 M</td>
<td>0%</td>
</tr>
<tr>
<td>GRID3</td>
<td>Production</td>
<td>3500</td>
<td>18</td>
<td>1.3 M</td>
<td>0%</td>
</tr>
</tbody>
</table>

*The Grid Workloads Archive: [http://gwa.ewi.tudelft.nl/pmwiki/]
Grid Workload Traces: Bursty Job Arrivals (5 minute intervals)

DAS3

SHARCNET

Bursty arrivals reduce predictability!
Job Runtime Predictions

Research Grids

Production Grids

Lower curves have higher accuracy

Job runtimes are predicted more accurately in research grids
Conclusion

• **Koala** supports multiple application types:
  • Parallel applications that may need co-allocation
  • Parallel applications that can grow/shrink at runtime
  • Parameter sweep applications
  • Workflows

• **Future Work:**
  • A decentralized P2P KOALA
  • Cloud resource management
Questions?

More Information:

- Koala: www.st.ewi.tudelft.nl/koala
- The Grid Workloads Archive: http://gwa.ewi.tudelft.nl/pmwiki/
- DGSim: www.pds.ewi.tudelft.nl/~iosup/dgsim.php
- see PDS publication database at: www.pds.twi.tudelft.nl/

email: o.o.sonmez@tudelft.nl

This work was carried out in the context of the Virtual Laboratory for e-Science project (www.vl-e.nl). Part of this work is also carried out under the FP6 Network of Excellence CoreGRID funded by European Commision.
Cycle Scavenging Policies

1. Equipartition-All (grid-wide basis)