Agent-Based Grid Load-Balancing

Daniel P. Spooner
University of Warwick, UK

Junwei Cao
NEC Europe Ltd., Germany
Outline

- Grid Computing & Middleware
- Agent-Based Methodology
- Distributed Service Discovery
- Grid Load-Balancing
- Experimental Results
- Conclusions & Future Work
Grid Computing

- Computational Grids - blueprint for a new computing infrastructure
- Data/Service/Community Grids - large-scale resource sharing in VOs
- Grid/Web Services - distributed system and application integration
Grid Middleware

- Resource Management & Scheduling
- Information Services (Monitoring & Discovery)
- Data Management and Access
- Application Programming Environments
- Security, Accounting, QoS ……

Condor, LSF, Ninf, Nimrod, ……
Globus, Legion, DPSS, ……
Java/Jini, CORBA, Web Services, ……
Grid Resource Management

Key challenges:
- Cross-domain
- Large-scale
- Dynamic
- QoS support
Agent-Based Methodology

An agent is:
- A local grid manager
- An user agent
- A broker
- A service provider
- A service requestor
- A matchmaker
Local Management

Performance Prediction
- Task models
- Hardware models

FIFO Algorithm
- Heuristic & Evolutionary
- Near-optimal on makespan, deadlines and idletime.

Processor 1
Processor 2
Processor 3
Processor 4
Processor 5
Processor 6
Processor 7
Processor 8

$2^{n-1}$
Service Discovery

- Pure data-pull
- No advertisement
- Full discovery
- Efficient when service change more quickly

- Pure data-push
- Full advertisement
- No discovery
- Efficient when requests arrive more frequently

Centralised, not applicable for grid computing!
Optimisation Strategies

- Configurable data-pull or data-push
- Agent hierarchy
- Multi-step advertisement & multi-step discovery
- Efficient when frequencies of request arrivals and service changes are almost the same

Distributed! balancing between advertisement and discovery!
Agent Implementation

- Resource Monitoring
- Service Info. Provider
- Database Management
- Advertisement

- Task Execution
- FIFO/GA Scheduling
- Task Management

- FIFO Scheduling
- Current HW Model
- Makespan

- Performance Prediction
- Task Model
- Discovery

- Match Maker
- Sched. Time
- User Deadline
- To Another Agent
Load Balancing Metrics

- Total makespan
- Average advance time of task execution completions (required deadline - actual task completion time)
- Average processor utilisation rate (busy time / total makespan)
- Load balancing level (1 - mean square deviation of processor utilisation rates / average processor utilisation rate)
- Total number of network packages for both advertisement and discovery
Experiment Design

Tasks:
sweep3d
fft
improc
closure
jacobi
memsort
cpi
Experiment 1

<table>
<thead>
<tr>
<th>Agent</th>
<th>Task Distribution</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>S01</td>
<td><img src="image1" alt="Task Distribution" /></td>
<td>7</td>
</tr>
<tr>
<td>S02</td>
<td><img src="image2" alt="Task Distribution" /></td>
<td>8</td>
</tr>
<tr>
<td>S03</td>
<td><img src="image3" alt="Task Distribution" /></td>
<td>5</td>
</tr>
<tr>
<td>S04</td>
<td><img src="image4" alt="Task Distribution" /></td>
<td>9</td>
</tr>
<tr>
<td>S05</td>
<td><img src="image5" alt="Task Distribution" /></td>
<td>10</td>
</tr>
<tr>
<td>S06</td>
<td><img src="image6" alt="Task Distribution" /></td>
<td>8</td>
</tr>
<tr>
<td>S07</td>
<td><img src="image7" alt="Task Distribution" /></td>
<td>7</td>
</tr>
<tr>
<td>S08</td>
<td><img src="image8" alt="Task Distribution" /></td>
<td>7</td>
</tr>
<tr>
<td>S09</td>
<td><img src="image9" alt="Task Distribution" /></td>
<td>11</td>
</tr>
<tr>
<td>S10</td>
<td><img src="image10" alt="Task Distribution" /></td>
<td>7</td>
</tr>
<tr>
<td>S11</td>
<td><img src="image11" alt="Task Distribution" /></td>
<td>8</td>
</tr>
<tr>
<td>S12</td>
<td><img src="image12" alt="Task Distribution" /></td>
<td>7</td>
</tr>
</tbody>
</table>
Experiment 2

<table>
<thead>
<tr>
<th>Agent</th>
<th>Task Distribution</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>S01</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>S02</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>S03</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>S04</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>S05</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>S06</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>S07</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>S08</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>S09</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>S10</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>S11</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>S12</td>
<td></td>
<td>7</td>
</tr>
</tbody>
</table>

Time:
- 0.0s
- 1099.0s
- 2198.0s
Experiment 3
Both GA and agents contribute towards the improvement in task executions.
Resource Utilisation

Less powerful S11 & S12 benefit mainly from the GA.

More powerful S1 & S2 benefit mainly from agents.
The GA contributes more to local grid load balancing.

Agents contribute more to global grid load balancing.
Total Makespan

The centralised pure data-pull can always achieve the best results.

Distributed agents with the hierarchical model can also achieve reasonably good results.
The network overhead for the pure data-pull strategy to achieve better results is very high.

Distributed agent-based service advertisement and discovery can scale well.
Conclusions

- An multi-agent paradigm provides a clear high-level abstraction of grid resource management system.
- Distributed service advertisement and discovery strategies can be used to improve agent performance.
- Agent-based framework is scalable, flexible, and extensible for further enhancements.