A Decoupled Scheduling Approach for the GrADS Program Development Environment

DCSL
Ahmed Amin
Outline

- Introduction
- Related Work
- Scheduling Architecture
- Scheduling Algorithm
- Testbench
- Results
- Conclusions
- Future work
Problem Definition

- Distributed parallel computation
- Grid Environment
  - Grid Application Development Software (GrADS) [3]
- Grids are heterogeneous systems that allow sharing of resources (cpu, mem, storage, n/w, ...)
- Grids are autonomous
Goal

- We require scheduling of application tasks onto available resources
- Minimize execution time of application
- **Decouple** scheduling from the application

  - Two main components:
    - *Performance model*: Analytical metric for predicting application execution times on a given set of resources
    - *Mapper*: Provides directives for mapping logical application data / tasks to physical resources
Related Work

- **AppLeS Project [4]:**
  - Embeds scheduling logic into application
  - Or embed application specific information into the scheduler
  - Not easily retargeted

- **Condor Matchmaker [5]**
  - Single processor

- **Prophet [6]**
  - Dynamic run-time scheduling on heterogeneous systems
  - Must be written in Mentat programming language
Related Work

- Meta-scheduler designs:
  - Application specific schedulers are used to determine schedules for each application
  - Meta-scheduler evaluates performance of multiple applications in the system
  - Meta-scheduler modifies the application specific schedules to improve performance
Scheduling Architecture

- MDS: Meta-computing Directory Service
  - (OS)
  - (CPU speed)
  - (# CPUs)

- NWS: Network Weather System
  - (% available CPU cycles)
  - (Free mem)
  - (BW)

An exhaustive search of p machines in the machine list \( \Rightarrow 2^p \)
Scheduling Algorithm

- **Schedule**: ordered list of machines and a mapping of data / tasks to those machines
- Desirable individual machine characteristics
- Desirable aggregate characteristics
  - E.g. fast machines with low network latency between them
Scheduling Algorithm

3p2s CMGs [2]

Algorithm: \textsc{ScheduleSearch}(machList, gridInfo, PerfModel, Mapper)

\begin{itemize}
  \item \texttt{sites} $\leftarrow$ \texttt{FindSites(machList)}
  \item \texttt{siteCollections} $\leftarrow$ \texttt{ComputeSiteCollections(sites)}
  \item For each \texttt{collection} in \texttt{siteCollections}
    \item For each \texttt{machineMetric} in (computation, memory, dual)
      \item For targetSize $\leftarrow$ 1 to \texttt{size(collection)}
        \item \texttt{list} $\leftarrow$ \texttt{SortDescending(collection, machineMetric)}
        \item \texttt{CMG} $\leftarrow$ \texttt{GetFirstN(list, targetSize)}
        \item \texttt{currSched} $\leftarrow$ \texttt{GenerateSchedule(CMG, Mapper, PerfModel)}
        \item If \texttt{currSched.predTime} $<$ \texttt{bestSched.predTime}
          \item \texttt{bestSched} $\leftarrow$ \texttt{currSched}
      \item \texttt{return (bestSched)}
\end{itemize}
Scheduling Information

- Obtained from MDS, NWS

- Static
- Dynamic
- User
Testbench

- Game of Life
- Jacobi
- Problem size N
- C & MPI / MPICH-G
- Performance Model: simple computation to predict iteration times

Mapper:
- Time-balance machine utilization (minimize idle times of selected machines)
- Constraint optimization $\rightarrow$ lp_solve
Testbed

- One-site testbed
  - \( N = \{600, 1200, 2400, 4800, 7200, 9600\} \)

- Three-site testbed
  - \( N = \{600, 4800, 9600, 14400, 16800, 19200\} \)

- NWS “up” vs NWS “down”
Experimental Scenarios

- Dynamic, Static, User
- 104 iterations per application
- 2 app * 2 testbeds * 6 input * 3 strategies

\[ \text{deg From Best} = \frac{\text{itTime} - \text{itTime}_{\text{best}}}{\text{itTime}_{\text{best}}} \times 100 \]
Results

![Bar chart showing percent degradation from best for different scenarios.]

- Game of Life One-site
- Game of Life Three-site
- Jacobi One-site
- Jacobi Three-site

Legend:
- User
- Static
- Dynamic
## Results

<table>
<thead>
<tr>
<th></th>
<th>Game of Life - 1 site</th>
<th></th>
<th>Game of Life - 3 site</th>
<th></th>
<th>Jacobi - 1 site</th>
<th></th>
<th>Jacobi - 3 site</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg</td>
<td>240.0</td>
<td>37.3</td>
<td>5.1</td>
<td>381.9</td>
<td>30.8</td>
<td>3.8</td>
<td>210.3</td>
<td>17.2</td>
</tr>
<tr>
<td>Std</td>
<td>152.0</td>
<td>40.4</td>
<td>12.9</td>
<td>466.6</td>
<td>63.3</td>
<td>10.7</td>
<td>130.6</td>
<td>28.2</td>
</tr>
<tr>
<td>Min</td>
<td>7.7</td>
<td>0</td>
<td>0</td>
<td>45.3</td>
<td>0</td>
<td>0</td>
<td>16.4</td>
<td>0</td>
</tr>
<tr>
<td>Max</td>
<td>507.7</td>
<td>156.9</td>
<td>69.3</td>
<td>2748.0</td>
<td>421.8</td>
<td>68.5</td>
<td>466.4</td>
<td>90.5</td>
</tr>
</tbody>
</table>
## Scheduling Overhead

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Info retr. (s)</th>
<th>Total (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local MDS, NWS server</td>
<td>2.0</td>
<td>4.5</td>
</tr>
<tr>
<td>Remote NWS server</td>
<td>59.6</td>
<td>62.4</td>
</tr>
<tr>
<td>Remote MDS server</td>
<td>1087.5</td>
<td>1088.4</td>
</tr>
</tbody>
</table>
Conclusion

- Application has been decoupled from scheduling
- Reduced execution time compared to user defined schedules
- Exploits dynamic information at run time for improved scheduling
Future Work

- Extend performance evaluation on parameters other than execution time
- Compiler generated performance models
- Mapping strategies
References

<table>
<thead>
<tr>
<th></th>
<th>Circus machines</th>
<th>Torc machines</th>
<th>Opus machines</th>
<th>Major machines</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>ucsd.edu</td>
<td>cs.utk.edu</td>
<td>cs.uiuc.edu</td>
<td>cs.uiuc.edu</td>
</tr>
<tr>
<td>Nodes</td>
<td>6</td>
<td>8</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Names</td>
<td>dralion, mystere soleil, quidam saltimbanco nouba</td>
<td>torc1, torc2 torc3, torc4 torc5, torc6 torc7, torc8</td>
<td>opus13-m opus14-m opus15-m opus16-m</td>
<td>amajor, bmajor cmajor, fmajor gmajor, hmajor</td>
</tr>
<tr>
<td>Processor</td>
<td>450 MHz PIII dralion, nouba 400 MHz PII others</td>
<td>550 MHz PIII</td>
<td>450 MHz PII</td>
<td>266 PII</td>
</tr>
<tr>
<td>CPUs/Node</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Memory/Node</td>
<td>256 MB</td>
<td>512 MB</td>
<td>256 MB</td>
<td>128 MB</td>
</tr>
<tr>
<td>OS</td>
<td>Debian Linux</td>
<td>Red Hat Linux</td>
<td>Red Hat Linux</td>
<td>Red Hat Linux</td>
</tr>
<tr>
<td>Kernel</td>
<td>2.2.19</td>
<td>2.2.15 SMP</td>
<td>2.2.16</td>
<td>2.2.19</td>
</tr>
<tr>
<td>Network</td>
<td>100 Mbps shared ethernet</td>
<td>100 Mbps switched Ethernet</td>
<td>100 Mbps switched Ethernet</td>
<td>100 Mbps shared ethernet</td>
</tr>
</tbody>
</table>
<> **What is Grid?**

In June'02, I attended the Grid Computing Planet conference in San Jose, California and I was surprised to learn that people even call clusters as grids. I believe that it is a marketing hype. Here is my definition of the Grid, which is based on my presentation as part of the "Understanding the Grid" panel:

**Grid is a type of parallel and distributed system that enables the sharing, selection, and aggregation of geographically distributed "autonomous" resources dynamically at runtime depending on their availability, capability, performance, cost, and users' quality-of-service requirements.**

It should be noted that Grids aim at exploiting synergies that result from cooperation--ability to share and aggregate distributed computational capabilities and deliver them as service.

**How is Grid different from other technologies such as Clusters/P2P/ASP? What about Grid Economy and Resource Management? What about your work in this area? and so on...**

I have been asked such questions very frequently, which has trigged me to start creating this FAQ page.

The key distinction between clusters and grids is mainly lie in the way resources are managed. In case of clusters, the resource allocation is performed by a centralised resource manager and all nodes cooperatively work together as a single unified resource. In case of Grids, each node has its own resource manager and don't aim for providing a single system view. Some of these points are being highlighted in my [panel presentation at P2P 2002 conference](#). It should be noted that autonomous resources in the Grid can span across a single or multiple organisations.