A New Parallel Asynchronous Cellular Genetic Algorithm for Mapping in Grids

Frédéric Pinel, Bernabé Dorronsoro, Pascal Bouvry

NIDISC 2010
Outline

- Contribution
- Problem description
- Algorithms
- Results
- Future work
Contribution

- Apply a new multi-core model for independent task scheduling on grids
- New local search operator
- Improve previous results
Problem description (1)

- Map heterogeneous independent tasks to heterogeneous machines
  - 512 tasks, 16 machines
- Expected Time to Compute (ETC) model
- Minimize makespan
- Limited execution time (90 s)
Problem description (2)

12 ETC instances used:

- $u_c_{hihi.0}$
- $u_s_{hihi.0}$
- $u_i_{hihi.0}$
- $u_c_{hilo.0}$
- $u_s_{hilo.0}$
- $u_i_{hilo.0}$
- $u_c_{lohi.0}$
- $u_s_{lohi.0}$
- $u_i_{lohi.0}$
- $u_c_{lolo.0}$
- $u_s_{lolo.0}$
- $u_i_{lolo.0}$
Algorithms (1)

- Cellular genetic algorithm
- Asynchronous
Algorithms (2)

Parallelism
Algorithms (3)

Representation

ETC

... machine i ... task i
machine i+1 task i+1
... ...
Algorithms (4)

• Representation

2759140368

• Crossover: 2 point cross-over

If Individual 2 has better fitness value
Algorithms (5)

Local search
- Select a random task from most loaded machine
- Move to one of the least loaded machines, whose new completion time is smallest
- Iterate
• Population: 16 x 16
• Initialize 1 individual with Min-Min
• Threads: 1-4
• Recombination: 1 or 2 point cross-over
• Mutation: move random task to random machine
• Local search iterations: 5-10
• Replace if better
• Processor: Xeon 2.8 GHz, 4 cores (2007)
Results (1)

Speed-up
Results (2)

• Recombination
• Local search iterations
## Results (3)

### Comparison of mean makespan

<table>
<thead>
<tr>
<th>instance</th>
<th>Struggle GA</th>
<th>CMA + LTH</th>
<th>PA-CGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>u_c_hihi.0</td>
<td>7,752,349.4</td>
<td>7,554,119.4</td>
<td>7,437,591.3</td>
</tr>
<tr>
<td>u_c_hilo.0</td>
<td>155,571.5</td>
<td>154,057.6</td>
<td>154,392.8</td>
</tr>
<tr>
<td>u_c_lohi.0</td>
<td>250,550.9</td>
<td>247,421.3</td>
<td>242,061.8</td>
</tr>
<tr>
<td>u_c_lolo.0</td>
<td>5,240.1</td>
<td>5,148.8</td>
<td>5,247.9</td>
</tr>
<tr>
<td>u_s_hihi.0</td>
<td>4,371,324.5</td>
<td>4,337,494.6</td>
<td>4,229,018.4</td>
</tr>
<tr>
<td>u_s_hilo.0</td>
<td>98,334.6</td>
<td>97,426.2</td>
<td>97,424.8</td>
</tr>
<tr>
<td>u_s_lohi.0</td>
<td>127,762.5</td>
<td>128,216.1</td>
<td>125,579.3</td>
</tr>
<tr>
<td>u_s_lolo.0</td>
<td>3,539.4</td>
<td>3,488.3</td>
<td>3,526.6</td>
</tr>
<tr>
<td>u_i_hihi.0</td>
<td>3,080,025.8</td>
<td>3,054,137.7</td>
<td>3,011,581.3</td>
</tr>
<tr>
<td>u_i_hilo.0</td>
<td>76,307.9</td>
<td>75,005.5</td>
<td>74,476.8</td>
</tr>
<tr>
<td>u_i_lohi.0</td>
<td>107,294.2</td>
<td>106,158.7</td>
<td>104,490.1</td>
</tr>
<tr>
<td>u_i_lolo.0</td>
<td>2,610.2</td>
<td>2,597.0</td>
<td>2,602.5</td>
</tr>
</tbody>
</table>
## Results (4)

### Comparison of mean makespan

<table>
<thead>
<tr>
<th>instance</th>
<th>Struggle GA</th>
<th>CMA + LTH</th>
<th>PA-CGA 10s</th>
<th>PA-CGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>u_c_hihi.0</td>
<td>7,752,349.4</td>
<td>7,554,119.4</td>
<td>7,518,600.7</td>
<td>7,437,591.3</td>
</tr>
<tr>
<td>u_c_hilo.0</td>
<td>155,571.5</td>
<td>154,057.6</td>
<td>154,963.6</td>
<td>154,392.8</td>
</tr>
<tr>
<td>u_c_lohi.0</td>
<td>250,550.9</td>
<td>247,421.3</td>
<td>245,012.9</td>
<td>242,061.8</td>
</tr>
<tr>
<td>u_c_lolo.0</td>
<td>5,240.1</td>
<td>5,148.8</td>
<td>5,261.4</td>
<td>5,247.9</td>
</tr>
<tr>
<td>u_s_hihi.0</td>
<td>4,371,324.5</td>
<td>4,337,494.6</td>
<td>4,277,497.3</td>
<td>4,229,018.4</td>
</tr>
<tr>
<td>u_s_hilo.0</td>
<td>98,334.6</td>
<td>97,426.2</td>
<td>97,841.6</td>
<td>97,424.8</td>
</tr>
<tr>
<td>u_s_lohi.0</td>
<td>127,762.5</td>
<td>128,216.1</td>
<td>126,397.9</td>
<td>125,579.3</td>
</tr>
<tr>
<td>u_s_lolo.0</td>
<td>3,539.4</td>
<td>3,488.3</td>
<td>3,535.0</td>
<td>3,526.6</td>
</tr>
<tr>
<td>u_i_hihi.0</td>
<td>3,080,025.8</td>
<td>3,054,137.7</td>
<td>3,030,250.8</td>
<td>3,011,581.3</td>
</tr>
<tr>
<td>u_i_hilo.0</td>
<td>76,307.9</td>
<td>75,005.5</td>
<td>74,752.8</td>
<td>74,476.8</td>
</tr>
<tr>
<td>u_i_lohi.0</td>
<td>107,294.2</td>
<td>106,158.7</td>
<td>104,987.8</td>
<td>104,490.1</td>
</tr>
<tr>
<td>u_i_lolo.0</td>
<td>2,610.2</td>
<td>2,597.0</td>
<td>2,605.5</td>
<td>2,602.5</td>
</tr>
</tbody>
</table>
Summary

- Parallel asynchronous CGA for multi-core
- Applied to independent task mapping on grids
- Evaluated on benchmark instances
- Improved most results
Future work

• Paper extension:
  – Experiment with more instances of each ETC class
  – Study performance of algorithm with # threads (outside runtime considerations)
  – Heuristics & population initialization
  – Heterogeneous algorithms (parameters)