Alkis Vazacopoulos
Robert Ashford
Optimization Direct Inc.
INFORMS Fall 2015

CPLex
Very Large Optimization Models
And parallel processing
Summary

- Large Scale Optimization
- The Curse of Dimensionality: Size Matters
- What can be done?
- Parallel Processing and Using Multiple Threads
- Optimization on Multiple Threads
- Example: Large Scale Scheduling and Design Models
- ODHeuristics: Heuristic Solution Software
- Parallel Heuristic Approach
- Heuristic Results on Scheduling Models
Large Scale Optimization

• Many models now solved routinely which would have been impossible ('unsolvable') a few years ago

• **BUT**: have super-linear growth of solving effort as model size/complexity increases

• **AND**: customer models keep getting larger
  • Globalized business has larger and more complex supply chain
  • Optimization expanding into new areas, especially scheduling
  • Detailed models easier to sell to management and end-users
The Curse of Dimensionality: Size Matters

- Super-linear solve time growth often supposed
- The reality is worse
- Few data sets available to support this
- Look at randomly selected sub-models of two scheduling models
  - Simple basic model
  - More complex model with additional entity types
  - Two hour time limit on each solve
  - 8 threads on 4 core hyperthreaded Intel i7-4790K
- See how solve time varies with integers after presolve
Simple model

Solution time in seconds vs. Number of integer entities
Why size matters

• Solver has to
  • (Presolve and) solve LP relaxation
  • Find and apply cuts
  • Branch on remaining infeasibilities (and find and apply cuts too)
  • Look for feasible solutions with heuristics all the while

• Simplex relaxation solves theoretically NP, but in practice effort increases between linearly and quadratic

• Barrier solver effort grows more slowly, but:
  • cross-over still grows quickly
  • usually get more integer infeasibilities
  • can’t use last solution/basis to accelerate

• Cutting grows faster than quadratic: each cut requires more effort, more cuts/round, more rounds of cuts, each round harder to apply.

• **Branching is exponential**: $2^n$ in number of (say) binaries $n$
What can be done?

- Decomposition
  - Solve smaller models
- Use ‘good’ formulations
  - As tight as possible
  - Avoid symmetry
- Settle for a good solution
  - Use heuristics
- Use more powerful hardware
  - Can usually make minor improvements by having a faster processor and memory
  - Big potential is from parallel processing
  - Use 4, 8, 12, 24,... or more threads on separate ‘processors’
Moore’s Law Still Holds

Source: © 2014, James Reinders, Intel, used with permission
Parallel Processing

- Raw CPU clock speed is not improving
  - Power consumed (and heat generated) rises according to the square of the speed
- Memory speed is not improving (much)
  - Fastest DIMMs are low capacity
- Can get wider registers: vectorize
  - Do several (up to 8) fp calculations at once on 512 bit register (SIMD)
  - Of limited use in sparse optimization
- Can fit more processors onto a single chip
## Intel Server Processors

Can have 2 or 4 (sockets) on single motherboard

<table>
<thead>
<tr>
<th>Core(s)</th>
<th>Threads</th>
<th>SIMD Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>64-bit</td>
<td>5100 series</td>
<td>128</td>
</tr>
<tr>
<td>5500 series</td>
<td>5600 series</td>
<td>128</td>
</tr>
<tr>
<td>Sandy Bridge EP</td>
<td>Ivy Bridge EP</td>
<td>128</td>
</tr>
<tr>
<td>Haswell EP</td>
<td>Knights Corner</td>
<td>128</td>
</tr>
<tr>
<td>Knights Landing</td>
<td>60+</td>
<td>128</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>12</th>
<th>18</th>
<th>61</th>
<th>60+</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>8</td>
<td>12</td>
<td>16</td>
<td>24</td>
<td>36</td>
<td>244</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>128</td>
<td>128</td>
<td>128</td>
<td>256</td>
<td>256</td>
<td>256</td>
<td>512</td>
<td></td>
</tr>
</tbody>
</table>
Using Multiple Threads

- Parallelize proportion $p$ of algorithm
- Amdahl’s law: speedup from $n$ threads is
  \[
  \frac{n}{n(1-p) + 1} < \frac{1}{1-p}
  \]
- Sets an obvious limit on benefit of parallel threads
- Doing more each clock cycle
- {fix, multiply, add} now quicker than reading and writing items to and from memory
- Memory access now the bottleneck (~20GB/s)
- Use out-of-order execution and SMT
Optimization on Multiple Threads

- CPLEX can use multiple threads for:
  - Root solve
    - probing
    - concurrent/barrier LP solve
  - cutting
  - heuristics
  - Branch & Cut
  - Can’t lock threads: avoid SMT (hyperthreading)
  - Heavy memory use
  - Work rate drops as use more than 4 threads/socket
  - Penalty for splitting threads across sockets
**Example: Large Scale Scheduling and Design Models**

<table>
<thead>
<tr>
<th>Model</th>
<th>entities</th>
<th>periods or boxes</th>
<th>rows</th>
<th>cols</th>
<th>integers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Easy</td>
<td>6362</td>
<td>5</td>
<td>299288</td>
<td>57804</td>
<td>57804</td>
</tr>
<tr>
<td>Medium</td>
<td>314</td>
<td>64</td>
<td>389560</td>
<td>94200</td>
<td>94200</td>
</tr>
<tr>
<td>Large</td>
<td>406</td>
<td>64</td>
<td>371964</td>
<td>149132</td>
<td>149132</td>
</tr>
<tr>
<td>Difficult</td>
<td>200</td>
<td>n/a</td>
<td>2577916</td>
<td>1294400</td>
<td>1294400</td>
</tr>
</tbody>
</table>

- No usable (say within 30% gap) solution to medium model after 3 days run time on fastest hardware (Intel i7 4790K ‘Devil's Canyon’).
Work Rate on Multiple Threads

Intel Xeon E5-2690v3 (12 cores x 2 NUMA nodes)

Clicks/thread/second vs. Threads

- Easy
- Medium
- Large
- Difficult
ODHeuristics: Heuristic Solution Software

- Proof of optimality may be impractical, but want good solutions to (say) 20%
- Uses CPLEX callable library
- First-feasible-solution heuristics
- Improvement heuristic
  - Solves sequence of smaller sub-model(s)
    - approach used e.g. by RINS and local branching
    - Use model structure to create sub-models and combine solutions
- Assess solution quality by a very aggressive root solve of whole model
- Multiple instances can be run concurrently with different seeds
- Can run on only one thread or multiple threads
- Can interrupt at any point and take best solution so far
  time limit / call-back / SIGINT
Parallel Heuristic Approach

- Run several heuristic threads with different seeds simultaneously
- CPLEX callable library very flexible, so
  - Exchange solution information between runs
  - Kill sub-model solves when done better elsewhere
  - Opportunistic
- Improves sub-model selection
- Not subject to Amdahl’s law
- instances run on 2 x 12 core Xeon E5-2690v3
  - Each heuristic thread run with single CPLEX thread
    i.e. 1 core each
Example: Large Scale Scheduling and Design Models

<table>
<thead>
<tr>
<th>Model</th>
<th>entities</th>
<th>periods or boxes</th>
<th>rows</th>
<th>cols</th>
<th>integers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Easy</td>
<td>6362</td>
<td>5</td>
<td>299288</td>
<td>57804</td>
<td>57804</td>
</tr>
<tr>
<td>Medium</td>
<td>314</td>
<td>64</td>
<td>389560</td>
<td>94200</td>
<td>94200</td>
</tr>
<tr>
<td>Large</td>
<td>406</td>
<td>64</td>
<td>371964</td>
<td>149132</td>
<td>149132</td>
</tr>
<tr>
<td>Difficult</td>
<td>200</td>
<td>n/a</td>
<td>2577916</td>
<td>12944400</td>
<td>12944400</td>
</tr>
</tbody>
</table>

- No usable (say within 30% gap) solution to medium model after 3 days run time on fastest hardware (Intel i7 4790K ‘Devil’s Canyon’)
# Heuristic Results on Scheduling Models

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>1</th>
<th>4</th>
<th>12</th>
<th>16</th>
<th>24</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threads</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Easy</td>
<td>time</td>
<td>234</td>
<td>84</td>
<td>185</td>
<td>132</td>
<td>210</td>
</tr>
<tr>
<td></td>
<td>sol</td>
<td>96</td>
<td>96</td>
<td>96</td>
<td>96</td>
<td>96</td>
</tr>
<tr>
<td>Medium</td>
<td>time</td>
<td>21189</td>
<td>23652</td>
<td>10528</td>
<td></td>
<td>30345</td>
</tr>
<tr>
<td></td>
<td>sol</td>
<td>116</td>
<td>114</td>
<td>114</td>
<td></td>
<td>114</td>
</tr>
<tr>
<td>Large</td>
<td>time</td>
<td>84424</td>
<td>18569</td>
<td>84465</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>sol</td>
<td>1558</td>
<td>1278</td>
<td>930</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Difficult</td>
<td>time</td>
<td>32891</td>
<td>41649</td>
<td>14346</td>
<td></td>
<td>23950</td>
</tr>
<tr>
<td></td>
<td>sol</td>
<td>423</td>
<td>393</td>
<td>358</td>
<td></td>
<td>362</td>
</tr>
</tbody>
</table>
### Heuristic Results on Scheduling Models

<table>
<thead>
<tr>
<th>Threads</th>
<th>1</th>
<th>4</th>
<th>12</th>
<th>16</th>
<th>24</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Easy</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>time</td>
<td>234</td>
<td>84</td>
<td>185</td>
<td>132</td>
<td>210</td>
</tr>
<tr>
<td>sol</td>
<td>96</td>
<td>96</td>
<td>96</td>
<td>96</td>
<td>96</td>
</tr>
<tr>
<td><strong>Medium</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>time</td>
<td>21189</td>
<td>23652</td>
<td>10528</td>
<td>30345</td>
<td></td>
</tr>
<tr>
<td>sol</td>
<td>116</td>
<td>114</td>
<td>114</td>
<td>114</td>
<td></td>
</tr>
<tr>
<td><strong>Large</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>time</td>
<td>19000</td>
<td>19000</td>
<td>19000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sol</td>
<td>1779</td>
<td>1278</td>
<td>994</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Difficult</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>time</td>
<td>15000</td>
<td>15000</td>
<td>15000</td>
<td>15000</td>
<td></td>
</tr>
<tr>
<td>sol</td>
<td>436</td>
<td>408</td>
<td>358</td>
<td>371</td>
<td></td>
</tr>
</tbody>
</table>
Easy Model Heuristic Behavior

![Graph](image)

- Solution value vs. Time in seconds with varying thread counts:
  - 1 thread
  - 4 threads
  - 8 threads
  - 12 threads
  - 16 threads
  - 24 threads
Medium Model Heuristic Behavior

Solution value vs. Time in seconds

- 1 thread
- 4 threads
- 12 threads
- 24 threads
Large Model Heuristic Behavior

![Graph showing solution value over time in seconds for 1 thread, 4 threads, and 12 threads. The y-axis represents solution value, ranging from 0 to 5000, and the x-axis represents time in seconds, ranging from 0 to 30000. The graph shows a downward trend as time increases, with different slopes for each thread count.](image-url)
Difficult Model Heuristic Behavior

Solution value vs Time in seconds for 1 thread, 4 threads, 12 threads, and 24 threads.
Parallel Heuristic Advantages

- Better objective value
- More consistent
- Faster
- Compare time to interesting (i.e. good) solutions
- Speedup depends on model (as with straight MIP)

<table>
<thead>
<tr>
<th>Threads</th>
<th>4</th>
<th>8</th>
<th>12</th>
<th>16</th>
<th>24</th>
</tr>
</thead>
<tbody>
<tr>
<td>Easy</td>
<td>2.8</td>
<td>2.6</td>
<td>2.7</td>
<td>1.8</td>
<td>1.1</td>
</tr>
<tr>
<td>Medium</td>
<td>1.1</td>
<td>3.4</td>
<td>1.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Large</td>
<td>66.1</td>
<td>99.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Difficult</td>
<td>6.1</td>
<td>14.5</td>
<td>6.6</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- A factor of 4 with 12 cores is typical but can be more
Thanks for listening

Robert Ashford
rwa@optimizationdirect.com
www.optimizationdirect.com