Crunch: Search-based Hierarchy Generation for State Machines

Mathew Hall
University of Sheffield
Background

- State machines
  - Reverse engineered
  - Transition label only, no state names
State Machines

Small & Manageable

Large and difficult

http://www.flickr.com/photos/gattou/2285955438/sizes/m/

http://www.flickr.com/photos/kevenlaw/2887820440/sizes/m/in/photostream/
Solution: Add a Hierarchy
Complexity Reduction

• Abstraction
  – Only look at what you want to see
• Cyclomatic complexity
  – $E - N + 2$
  – $E$ is reduced when a transition from a group replaces individual transitions
Hierarchy Generation = Clustering

• Bundle related states together
• What does ‘related’ mean?
  – Generally
  – Given limited information available?
Bunch: Search-Based Clustering

- Classic example: Bunch [Mitchell & Mancoridis]
- Similar problem
  - Software module dependency graph (MDG)
  - Group similar modules together

An example MDG
Bunch: Search-Based Clustering

- Multiple Hill Climbs
- Maximises a fitness function
  - MQ
  - Minimise number of edges between groups
  - Designed with low coupling, high cohesion in mind
Crunch: Search Based State Machine Clustering

- Bunch works (very well!) but isn’t ideal for state machines
- Crunch mitigates problems with Bunch:
  - Takes a state machine (reflexive edges and all)
  - GA, evolves a linear sequence of merges
  - Designed with fitness function development in mind
- Open source: http://code.google.com/p/sbgct
Crunch

Getting results, but how good are they?
Evaluating Hierarchies

• Generally:
  – Compare with a known good solution

• Bunch:
  – Stability: compare fitness values of many solutions
  – Ask developers

• These mostly assess how good the algorithm is at approximating existing results, not how good the result is
Evaluating Crunch Output

• No ‘right’ answer
• No expert to discuss a result with
• Currently just using a set of metrics:
  – Cyclomatic complexity
    • Flat state machine after merging edges
    • Top level
  – Number of edges between & within clusters
  – MQ
Work so far...

- Crunch
- Technique for getting state machines via AspectJ & StateChum
- Fitness functions:
  - MQ
  - Similarity (based on connectivity of two states)
  - Cyclomatic complexity
  - and more...
# Results: Commons HashBag

<table>
<thead>
<tr>
<th>Fitness Function</th>
<th>Intra Edges</th>
<th>Extra Edges</th>
<th>Top Level CC</th>
<th>Number of Clusters</th>
<th>MQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>MQ</td>
<td>Avg.</td>
<td>23.1</td>
<td>14.9</td>
<td>9.8</td>
<td>7.1</td>
</tr>
<tr>
<td></td>
<td>Min</td>
<td>19</td>
<td>10</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>28</td>
<td>19</td>
<td>15</td>
<td>9</td>
</tr>
<tr>
<td>Cyclomatic</td>
<td>Avg.</td>
<td>9.1</td>
<td>28.9</td>
<td>23.9</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>Min</td>
<td>5</td>
<td>23</td>
<td>20</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>15</td>
<td>33</td>
<td>26</td>
<td>9</td>
</tr>
<tr>
<td>FeatureSet</td>
<td>Avg.</td>
<td>28.9</td>
<td>9.1</td>
<td>8.6</td>
<td>2.5</td>
</tr>
<tr>
<td></td>
<td>Min</td>
<td>20</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>37</td>
<td>18</td>
<td>18</td>
<td>3</td>
</tr>
<tr>
<td>Lutz</td>
<td>Avg.</td>
<td>23.2</td>
<td>14.8</td>
<td>13.9</td>
<td>2.9</td>
</tr>
<tr>
<td></td>
<td>Min</td>
<td>9</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>36</td>
<td>29</td>
<td>27</td>
<td>4</td>
</tr>
<tr>
<td>XOR</td>
<td>Avg.</td>
<td>9.5</td>
<td>28.5</td>
<td>28.5</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Min</td>
<td>6</td>
<td>26</td>
<td>26</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>12</td>
<td>32</td>
<td>32</td>
<td>2</td>
</tr>
<tr>
<td>Random</td>
<td>Avg.</td>
<td>18</td>
<td>20</td>
<td>18.4</td>
<td>3.6</td>
</tr>
<tr>
<td></td>
<td>Min</td>
<td>10</td>
<td>4</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>34</td>
<td>28</td>
<td>26</td>
<td>6</td>
</tr>
</tbody>
</table>
Future Work

• Improve evaluation method
  – Better metrics
  – human-based study
• Develop fitness functions
• In-depth case studies