Managing Uncertainty in Self-* Systems with Plan Reuse and Stochastic Search

Cody Kinneer, Zack Coker, Joanna Wang, David Garlan, Claire Le Goues
Cloud Web Server

Data Center Properties
- Server type
- Traffic level
- Dimmer
- Number of servers

Server Properties
- Cost
- Max full requests
- Max dimmed requests
Carnegie Mellon University

Cloud Web Server

Users

Data Center A

Data Center B

Data Center C
Tactics

- Start Server
- Shutdown Server
- Increase Dimmer
- Decrease Dimmer
- Increase Traffic
- Decrease Traffic
Time and Failure

Start Server
Existing Approaches

- Rainbow / Stitch
- Planning
  - Manual
  - Automated
  - PRISM

Handle Evolution?
Evolution

- Evolving Tactics
- Evolving Environment
- Evolving Quality Priorities
Our Approach

• Genetic Programming
  • Inspired by automated repair
• Simple planning language
• Reuse enabling techniques
• Implemented in Java using ECJ library from George Mason University
\langle plan \rangle ::= \mathtt{'} ( \langle operator \rangle \ mathtt{'}) \mathtt{'} \\
| \mathtt{'} ( \langle tactic \rangle \ mathtt{'}) \mathtt{'} \\

\langle operator \rangle ::= \langle for-loop \rangle \\
| \langle try-catch \rangle \\
| \langle sequence \rangle \\

\langle for-loop \rangle ::= \mathtt{F} \langle int \rangle \langle plan \rangle \\

\langle int \rangle ::= \mathtt{[2-10]} \\

\langle sequence \rangle ::= \mathtt{;} \langle plan \rangle \langle plan \rangle \\

\langle try-catch \rangle ::= \mathtt{T} \langle plan \rangle \langle plan \rangle \langle plan \rangle
Plans

;  

F  

5  StartServer  

C  

F  

4  StartServer  

B
Plan Evaluation

StartServer A

0.10

StartServer A

0.90

StartServer A

987.8

StartServer B

1137.3

StartServer B

1137.3

1526.6
Good Plans are Big
Reuse Enabling

- Kill ratio
- Scratch ratio
- Plan Trimmer
Kill Ratio
Scratch Ratio
Empirical Evaluation

- Does plan reuse result in improved fitness?
- Change model and plan for the new situation
- All results involving randomness are the median of 10 trials
- P values obtained by Wilcoxon rank sum test
Request Spike & New Data Center
## Reuse Enabling Techniques

<table>
<thead>
<tr>
<th>Planning Technique</th>
<th>Utility</th>
<th>P Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scratch</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>Scratch &amp; Kill Ratio</td>
<td>1.044</td>
<td>&lt; 0.01</td>
</tr>
<tr>
<td>Naive Reuse</td>
<td>0.962</td>
<td>0.06</td>
</tr>
<tr>
<td>Reuse &amp; Kill Ratio</td>
<td>1.072</td>
<td>&lt; 0.01</td>
</tr>
<tr>
<td>Reuse &amp; Kill Ratio &amp; Scratch Ratio</td>
<td>1.077</td>
<td>0.63</td>
</tr>
<tr>
<td>Reuse &amp; Kill Ratio &amp; Scratch Ratio &amp; Trimmer</td>
<td>1.112</td>
<td>&lt; 0.01</td>
</tr>
</tbody>
</table>
Request Spike & New Data Center

Profit

Generation

Starting Plan
reuse scratch
## Reuse Improvement

<table>
<thead>
<tr>
<th>Scenario</th>
<th>1k</th>
<th>10k</th>
</tr>
</thead>
<tbody>
<tr>
<td>Increased Costs</td>
<td>0.02</td>
<td>0.81</td>
</tr>
<tr>
<td>Network Unreliability</td>
<td>0.01</td>
<td>0.10</td>
</tr>
<tr>
<td>Failing Data Center</td>
<td>-0.02</td>
<td>0.14</td>
</tr>
<tr>
<td>Request Spike</td>
<td>-0.14</td>
<td>-0.01</td>
</tr>
<tr>
<td>New Data Center</td>
<td>-0.63</td>
<td>0.28</td>
</tr>
<tr>
<td>Request Spike &amp; New Data Center</td>
<td>-0.47</td>
<td>1.54</td>
</tr>
</tbody>
</table>
Key Contributions

- A planner based on genetic programming for self-adaptive systems
- A set of techniques for enabling more efficient plan reuse in GP
- Enhancing existing plans can result in improved fitness compared to planning from scratch

Cody Kinneer  ckinneer@andrew.cmu.edu
Adapting Plan to New Tactic
# Empirical Results

<table>
<thead>
<tr>
<th>Planning Technique</th>
<th>Utility</th>
<th>P Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scratch</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>Reuse</td>
<td>0.962</td>
<td>0.06</td>
</tr>
</tbody>
</table>
Evolving Tactics

Start Server @ D
Carnegie Mellon University

Cloud Web Server

Users

Data Center A

Data Center B

Data Center C
Empirical Evaluation

- Sanity check: compare planner to PRISM probabilistic model checker
- Is the planner close to optimal?
- How much faster?