Policy Learning for Adaptive Allocation of Cloud Resources to Multi-tier Web Applications

Waheed Iqbal,¹ Matthew N. Dailey,¹ David Carrera²
¹Asian Institute of Technology (AIT) Thailand, ²Technical University of Catalonia (UPC) Spain.

Motivation
Response time is an important quality attribute of a Web application.

Offering response time guarantees for multi-tier Web applications is challenging:
- a bottleneck tier affects all dependent tiers
- bottleneck locations depend on workload patterns
- the workload pattern is dynamic

RUBiS benchmark application saturates at different workload levels under different workload patterns:

Motivation
We propose a simple method to learn workload specific resource allocation policies to offer response time guarantees using minimal resources.

Workload Pattern Modeling
The workload pattern modeling method learns a clustering model:

![Workload Pattern Model](image)

For a specific time interval, the workload pattern is a vector containing the probability of each URI cluster.

Policy Learning
We use a greedy Q-learning method based on online observation.

The system state consists of:
- the tier configuration
- the current workload pattern vector
- the current arrival rate
- the current response time

The environment that the agent interacts with is a stochastic function mapping the current state and action to a new state.

The reward function:
- encourages the agent on success
- discourages the agent on failure to maintain the SLA
- discourages overprovisioning

Experimental Evaluation
- EUCALYPTUS-based private testbed cloud
- Two-tier RUBiS benchmark Web application
- Two experiments executed in three phases:
  - Exploration: we initialize an empty policy and let the system learn in real time
  - Exploitation: the agent simply uses the previously-learned policy to automatically resolve bottlenecks
  - Baseline: scale up every replicable tier every time a bottleneck occurs

Experimental Evaluation
Experiment 1 Exploitation Phase. Fewer CPU hours, small increase in SLA violations.

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Total allocated CPU hours</th>
<th>Percentage of requests violating SLA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>17.7</td>
<td>1.03</td>
</tr>
<tr>
<td>Exploitation</td>
<td>19.8</td>
<td>0.35</td>
</tr>
<tr>
<td>Baseline</td>
<td>24.7</td>
<td>1.75</td>
</tr>
<tr>
<td>Exploitation</td>
<td>27.0</td>
<td>0.233</td>
</tr>
</tbody>
</table>

Conclusion and Future Work
The proposed approach:
- enables us to learn on-line autoscaling policies for multi-tier Web applications
- helps us to offer response time guarantees
- minimizes resource allocation
- does not require any prior knowledge of the application
- minimizes the overhead needed to monitor, detect, and resolve bottlenecks

We are currently:
- improving the exploration phase
- integrating scale-down policy learning