BUNGEEL
An Elasticity Benchmark for Self-Adaptive IaaS Cloud Environments

Andreas Weber, Nikolas Herbst, Henning Groenda, Samuel Kounev

Munich - November 5th, 2015
Symposium on Software Performance 2015

http://descartes.tools/bungee
Characteristics of ...

Rubber Bands
- Base Length
- Width / Thickness / Force
- Stretchability
- Elasticity

IaaS Clouds
- Performance (1 resource unit)
- Quality Criteria / SLA
- Scalability
- Elasticity

Graphs showing size over time for Rubber Bands and IaaS Clouds.
Comparing Elastic Behavior of ...

Rubber Bands

Measure Elasticity
Independent of Performance and Scalability

IaaS Clouds
Motivation & Related Work
Why measure Elasticity?

Elasticity

- Major quality attribute of clouds  
  [Gartner09]
- Many strategies exist  
  [Galante12, Jannings14]
- Industry
- Academia

“**You can't control what you can't measure?**” (DeMarco)
“**If you cannot measure it, you cannot improve it**” (Lord Kelvin)
Related Work: Elasticity Benchmarking Approaches

- **Specialized approaches**
  - Measure technical provisioning time
  - Measure SLA compliance
  - Focus on scale up/out

- **Business perspective**
  - What is the financial impact?
  - Disadvantage: Mix-up of elasticity technique and business model

---

[Binning09, Li10, Dory11, Almeida13]
[Weimann11, Folkerts12, Islam12, Moldovan13, Tinnefeld14]
Concept & Implementation
Elasticity Benchmarking Concept

- System Analysis
- Benchmark Calibration
- Measurement
- Elasticity Evaluation

Analyze the performance of underlying resources & scaling behavior
Analyze System

■ Approach
  ■ Evaluate system separately at each scale
  ■ Find maximal load intensity that the system can withstand without violating SLOs (binary search)
  ■ Derive demand step function:
    \[ \text{resourceDemand} = f(\text{intensity}) \]

■ Benefit
  ■ Derive resource demand for arbitrary load intensity variations
Elasticity Benchmarking Concept

System Analysis

Benchmark Calibration

Measurement

Elasticity Evaluation

Analyze the performance of underlying resources & scaling behavior

Adjust load profile
Benchmark Calibration

- Approach: Adjust load intensity profile to overcome
  - Different performance of underlying resources
  - Different scalability
Elasticity Benchmarking Concept

System Analysis

Benchmark Calibration

Measurement

Elasticity Evaluation

- Analyze the performance of underlying resources & scaling behavior
- Adjust load profile
- Expose cloud system to varying load & monitor resource supply & demand
Measurement

Requirement: Stress SUT in a representative manner
- Realistic variability of load intensity
- Adaptability of load profiles to suit different domains

Approach:
- Open workload model [Schroeder06]
- Model Load Variations with the LIMBO toolkit [SEAMS15Kistowski]
  - Facilitates creation of new load profiles
    - Derived from existing traces
    - With desired properties (e.g. seasonal pattern, bursts)
- Execute load profile using JMeter
  - Timer-Plugin delays requests according to timestamp file created by LIMBO

http://descartes.tools/limbo

https://github.com/andreaswe/JMeterTimestampTimer
Elasticity Benchmarking Concept

- System Analysis
- Benchmark Calibration
- Measurement
- Elasticity Evaluation

1. Analyze the performance of underlying resources & scaling behavior
2. Adjust load profile
3. Expose cloud system to varying load & monitor resource supply & demand
4. Evaluate elasticity aspects accuracy & timing with metrics
Metrics: Accuracy (1/3)

accuracy_U: \( \frac{\sum U}{T} \)

accuracy_O: \( \frac{\sum O}{T} \)
Metrics: Timeshare (2/3)

timeshare_U: \[ \frac{\sum A}{T} \]

timeshare_O: \[ \frac{\sum B}{T} \]
Metrics: Jitter (3/3)

jitter: \[ \frac{E_S - E_D}{T} \]

\(E_D\): # demand adaptations, \(E_S\): # supply adaptations
Elasticity Benchmarking Concept

System Analysis

Benchmark Calibration

Measurement

Elasticity Evaluation

Analyze the performance of underlying resources & scaling behavior

Adjust load profile

Expose cloud system to varying load & monitor resource supply & demand

Evaluate elasticity aspects accuracy & timing with metrics
BUNGEE Implementation

- Java-based elasticity benchmarking framework

**Components**
- Harness (Benchmark Node)
- Cloud-side load generation application (CSUT)

**Automates the four benchmarking activities**
- System Analysis
- Benchmark Calibration
- Measurement
- Elasticity Evaluation

**Analysis of horizontally scaling clouds based on**
- CloudStack
- AWS
- [http://descartes.tools/bungee](http://descartes.tools/bungee)

**Extensible with respect to**
- new cloud management software
- new resource types
- new metrics

- Code is Open Source
- Quick Start Guide available
Evaluation & Case Study
Evaluation & Case Study

■ Evaluation (private cloud)
  ■ Reproducibility of system analysis
    \( \text{Err}_{rel} < 5\% \), confidence 95\% for first scaling stage
  ■ Simplified system analysis
    Linearity assumption holds for test system
  ■ Consistent ranking by metrics
    Separate evaluation for each metric, min. 4 configurations per metric

■ Case Study (private & public cloud)
  ■ Applicability in real scenario
  ■ Different performance of underlying resources
  ■ Metric Aggregation
Case Study: Configuration F - 1Core

<table>
<thead>
<tr>
<th>Configuration</th>
<th>accuracy₀ [res. units]</th>
<th>accuracyᵤ [res. units]</th>
<th>timeshare₀ [%]</th>
<th>timeshareᵤ [%]</th>
<th>jitter [adap/min.]</th>
<th>elastic speedup</th>
<th>violations [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>F – 1Core</td>
<td>2.423</td>
<td>0.067</td>
<td>66.1</td>
<td>4.8</td>
<td>-0.067</td>
<td>1.046</td>
<td>7.6</td>
</tr>
</tbody>
</table>
Case Study: Config. F - 2Core not adjusted

<table>
<thead>
<tr>
<th>Configuration</th>
<th>accuracy₀ [res. units]</th>
<th>accuracyₚ [res. units]</th>
<th>timeshare₀ [%]</th>
<th>timeshareₚ [%]</th>
<th>jitter [adapt/min.]</th>
<th>elastic speedup</th>
<th>violations [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>F – 1Core</td>
<td>2.423</td>
<td>0.067</td>
<td>66.1</td>
<td>4.8</td>
<td>-0.067</td>
<td>1.046</td>
<td>7.6</td>
</tr>
<tr>
<td>F – 2Core no adjustment</td>
<td>1.811</td>
<td>0.001</td>
<td>63.8</td>
<td>0.1</td>
<td>-0.033</td>
<td>1.291</td>
<td>2.1</td>
</tr>
</tbody>
</table>
## Case Study: Config. F - 2Core adjusted

<table>
<thead>
<tr>
<th>Configuration</th>
<th>accuracy₀ [res. units]</th>
<th>accuracyᵤ [res. units]</th>
<th>timeshare₀ [%]</th>
<th>timeshareᵤ [%]</th>
<th>jitter [adapt/min.]</th>
<th>elastic speedup</th>
<th>violations [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>F – 1Core</td>
<td>2.423</td>
<td>0.067</td>
<td>66.1</td>
<td>4.8</td>
<td>-0.067</td>
<td>1.046</td>
<td>7.6</td>
</tr>
<tr>
<td>F – 2Core no adjustment</td>
<td>1.811</td>
<td>0.001</td>
<td>63.8</td>
<td>0.1</td>
<td>-0.033</td>
<td>1.291</td>
<td>2.1</td>
</tr>
<tr>
<td>F – 2Core adjusted</td>
<td>2.508</td>
<td>0.061</td>
<td>67.1</td>
<td>4.5</td>
<td>-0.044</td>
<td>1.025</td>
<td>8.2</td>
</tr>
</tbody>
</table>
Conclusion

- Goal: Evaluate elastic behavior independent of
  - Performance of underlying resources and scaling behavior
  - Business model

- Contribution:
  - Elasticity benchmark concept for IaaS cloud platforms
  - Refined set of elasticity metrics
  - Concept implementation: BUNGEE - framework for elasticity benchmarking

- Evaluation:
  - Consistent ranking of elastic behavior by metrics
  - Case study on AWS and CloudStack

- Future Work:
  - BUNGEE: Distributed load generation, scale vertically, dif. resource types
  - Experiments: Tuning of elasticity parameters, evaluate proactive controllers

http://descartes.tools/bungee
Literature (1/2)


