Load Testing Elasticity and Performance Isolation in Shared Execution Environments

Samuel Kounev

Chair of Software Engineering
University of Würzburg
http://se.informatik.uni-wuerzburg.de/

Keynote talk, LT 2015 @ ICPE 2015, Austin, USA, Feb 1, 2015
Explosion of IT Service Clients
Growing Data Centers

Maiden, North Carolina (Apple)
46 000 m²

San Antonio (Microsoft)
43 000 m²

Prineville, Oregon (Facebook)
28 000 m²

Chicago (Digital Realty)
100 000 m²
Growing Number of Servers

- Google ~ 1 Mil. (2013)
- Microsoft ~ 1 Mil. (2013)
- Facebook ~ 180K (2012)
- OVH ~ 150K (2013)
- Akamai Tech. ~ 127K (2013)
- Rackspace ~ 94K (2013)
- 1&1 Internet ~ 70K (2010)
- eBay ~ 54K (2013)
- HP/EDS ~ 380K (2013)
- ...

Source: http://www.datacenterknowledge.com
Increasing Pressure to Raise Efficiency

- Proliferation of shared execution environments
- Different forms of resource sharing (hardware and software)
  - Network, storage, and computing infrastructure
  - Software stacks

Datacenter Sharing  Virtualization  Shared Middleware  Multi-Tenancy
Challenges

- Load Spike
- SLAs
- Expand / shrink resources on-the-fly

- When exactly should a reconfiguration be triggered?
- Which particular resources should be scaled?
- How quickly and at what granularity?
Consequences

- Increased system complexity and dynamics
- Diverse vulnerabilities due to resource sharing
- Inability to provide availability and performance guarantees
  \[\Rightarrow \text{Major distinguishing factor between service offerings}\]
- Lack of reliable benchmarks and metrics

“\textit{You can’t control what you can’t measure?}” (DeMarco)

“\textit{If you cannot measure it, you cannot improve it}” (Lord Kelvin)
Descartes Tool Chain

http://descartes.tools
Related Tools

- **BUNGEE** – Elasticity benchmarking framework ([homepage](#), [publications](#))
- **LIMBO** – Load intensity modeling tool ([homepage](#), [publications](#))
- **WCF** – Workload classification & forecasting tool ([homepage](#), [publications](#))
- **LibReDE** - Library for resource demand estimation ([homepage](#), [publications](#))
- **hInjector** – Security benchmarking tool ([homepage](#), [publications](#))
- **DML** – Descartes Modeling Language ([homepage](#), [publications](#))
- **DML Bench** ([homepage](#), [publications](#))
- **DQL** – Declarative performance query language ([homepage](#), [publications](#))
- **Further relevant research**
  - [http://descartes-research.net/research/research_areas/](http://descartes-research.net/research/research_areas/)
  - Self Aware Computing ([publications](#))

10 S. Kounev
The Focus of this Talk

Metrics and benchmarks for quantitative evaluation of

1. Resource elasticity
2. Performance isolation

in shared execution environments

- Virtualized infrastructures
- Multi-tenant applications
Credits

Nikolas Herbst + MSc students (elasticity)

Rouven Krebs + MSc students (performance isolation)
Part I: Resource Elasticity

Main references


Further references


What People Say Elasticity is...

OCDA [1]
- up & down scaling
- subscriber workload

NIST [2]
- rapid elasticity
- unlimited provision & release
- sometimes automated
- with demand

IBM, Schouten [3]
- scalability
- increase & reduce
- no manual labor

Eukalyptus, Wolski [4]
- measurable
- mapping of requests to resources

Cohen [5]
- quantifyable
- real-time demands
- local & remote
What is the relationship between the term elasticity (E) and the more classical term scalability (S)?

- A: E is a modern buzzword for S
- B: E is a prerequisite for S
- C: S is a prerequisite for E
- D: The terms are orthogonal
Elasticity vs. Scalability

What is the relationship between the term **elasticity** (E) and the more classical term **scalability** (S)?

- **A:** E is a modern buzzword for S
- **B:** E is a prerequisite for S
- **C:** S is a prerequisite for E
- **D:** The terms are orthogonal

**S. Kounev**

Resource Elasticity  
Performance Isolation  
Conclusions
Elasticity

Workload intensity (e.g., # requests / sec)

Service Level Objective (SLO)
(e.g., resp. time ≤ 2 sec, 95%)

Resource Demand
Minimal amount of resources required to ensure SLOs.

Amount of resources (e.g., # VMs)

Resource demand
underprovisioning
resource supply
overprovisioning
Def: The degree to which a system is able to adapt to workload changes by provisioning and deprovisioning resources in an autonomic manner, such that at each point in time the available resources match the current demand as closely as possible.

N. Herbst, S. Kounev and R. Reussner
Elasticity: What it is, and What it is Not.

Metrics: Accuracy

(1) accuracy$_U$: $\frac{\sum U}{T}$

(2) accuracy$_O$: $\frac{\sum O}{T}$

- red: res. demand
- blue: res. supply
(3) timeshare_U: \( \frac{\sum A}{T} \)  
(4) timeshare_O: \( \frac{\sum B}{T} \)
Metrics: Jitter

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

\( E_D \): # demand changes
\( E_S \): # supply changes
Elasticity Benchmarking

<table>
<thead>
<tr>
<th>Resource demand</th>
<th>Resource supply</th>
<th>Overprovisioning</th>
</tr>
</thead>
</table>

Same user workload on system B
System B at a doubled user workload

Resource Elasticity
Performance Isolation
Conclusions
Elasticity Benchmarking Approach

- **System Analysis**: Analyze efficiency & scaling behavior of underlying resources
- **Benchmark Calibration**: Adjust load profile
- **Measurement**: Expose SUT to varying load & monitor resource supply & demand
- **Metric Evaluation**: Compute elasticity metrics (accuracy & timing)

Step 2: Benchmark Calibration

- **Goal**: Induce same resource demand on all systems

- **Approach**: Adjust load intensity profile to overcome
  - Different efficiency of underlying resources
  - Different scalability
LIMBO: A Tool For Modeling Variable Load Intensities


Example: Wikipedia Workload

![DLIM_wikipedia Arrival Rates](chart.png)
Case Study: CloudStack (CS) - 1Core

CloudStack Settings
- quietTime: 120s
- condTrueDur: 30s
- threshUp: 65%
- threshDown: 10%

<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>CS – 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>
CloudStack (CS) – 2 Core – no adjustment

CloudStack Settings
- quietTime: 120s
- condTrueDur: 30s
- threshUp: 65%
- threshDown: 10%

<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>CS – 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>CS – 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>

S. Kounev

Resource Elasticity
Performance Isolation
Conclusions
CloudStack (CS) – 2 Core – adjusted

Configuration | accuracy₀ [res. units] | accuracyᵤ [res. units] | timeshare₀ [%] | timeshareᵤ [%] | jitter [adap/min.] | elastic speedup | violations [%]
--- | --- | --- | --- | --- | --- | --- | ---
CS – 1Core | 2.423 | 0.067 | 66.1 | 4.8 | -0.067 | 1.046 | 7.6
CS – 2Core no adjustment | 1.811 | 0.001 | 63.8 | 0.1 | -0.033 | 1.291 | 2.1
CS – 2Core adjusted | 2.508 | 0.061 | 67.1 | 4.5 | -0.044 | 1.025 | 8.2
Amazon Web Services (AWS) - m1.small

CloudStack Settings
- quietTime: 60s
- condTrueDur: 60s
- threshUp: 80%
- threshDown: 50%
- instUp/Down: 3/1

<table>
<thead>
<tr>
<th>Configuration</th>
<th>$\text{accuracy}_O$ [res. units]</th>
<th>$\text{accuracy}_U$ [res. units]</th>
<th>timeshare$_O$ [%]</th>
<th>timeshare$_U$ [%]</th>
<th>Jitter [adapt/min.]</th>
<th>Elastic Speedup</th>
<th>Violations [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS – 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>CS – 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>
<tr>
<td>AWS - m1.small</td>
<td>1.340</td>
<td>0.019</td>
<td>61.6</td>
<td>1.4</td>
<td>0.000</td>
<td>1.502</td>
<td>2.5</td>
</tr>
</tbody>
</table>
Part II: Performance Isolation

Main references


Further references


Example Scenario: Multi-Tenant Environments

Tenants working within their assigned quota (e.g., # users) should not suffer from tenants exceeding their quotas.
D is a set of **disruptive tenants** exceeding their quotas.

A is a set of **abiding tenants** not exceeding their quotas.

**Approach:** Quantify impact of increasing workload of the disruptive tenants on the performance of the abiding ones.
Metrics Based on QoS Impact

Reference Workload $W_{\text{ref}}$

Load

Tenants: t1, t2, t3, t4

Disruptive Workload $W_{\text{disr}}$

Load

Tenants: t1, t2, t3, t4

Load

Response time

Avg. response time for abiding tenants A

$W_{\text{ref}}$, $W_{\text{disr}}$

Workload

Different Response Times

$\Delta w = \frac{\sum_{t \in W_{\text{disr}}} w_t - \sum_{t \in W_{\text{ref}}} w_t}{\sum_{t \in W_{\text{ref}}} w_t}$

$\Delta z_A = \frac{\sum_{t \in A} [z_t(W_{\text{disr}}) - z_t(W_{\text{ref}})]}{\sum_{t \in A} z_t(W_{\text{ref}})}$
Example Metric

$$I_{QoS} = \frac{\Delta z}{\Delta w}$$

Difference in response time
Difference in workload

Perfectly Isolated = 0
Non-Isolated = ?

Answers: How strong is a tenant’s influence on the others?
Metrics Based on Workload Ratio

Workload vs. Time

Response time vs. Time

Resource Elasticity

Performance Isolation

Conclusions
For a given intensity of the disruptive workload, we plot the maximum possible intensity of the abiding workload, under which the QoS of the abiding tenants is maintained.
Metrics Based on Workload Ratio

We can maintain the QoS for the abiding tenant without decreasing his workload.
Metrics Based on Workload Ratio

Abiding workload

Isolated

Observed system

Non-isolated

\( W_{a,\text{ref}} \)

\( W_{a,\text{base}} \)

\( W_{d,\text{ref}} \)

\( W_{d,\text{base}} \)

\( W_{d,\text{end}} \)

Disruptive workload
Example Metric: $I_{end}$

$$I_{end} = \frac{W_{d_{end}} - W_{d_{base}}}{W_{a_{ref}}}$$

Perfectly Isolated = ?
Non-Isolated = 0

Answers: How isolated is the system compared to a non-isolated system?
Metrics Based on Workload Ratio Integral

- **Abiding Workload**
  - $W_{a_{\text{ref}}}$
  - $W_{a_{\text{base}}}$

- **Disruptive Workload**
  - $W_{d_{\text{ref}}}$
  - $W_{d_{\text{base}}}$
  - $W_{d_{\text{end}}}$

- **Non-Isolated**
  - $A_{\text{measured}}$

- **Isolated**

**Observed System**

**Performance Isolation**
Metrics Based on Workload Ratio Integrals

Abiding Workload

W_{a_{\text{ref}}}

W_{a_{\text{base}}}

A_{\text{nonIsolated}}

Non-Isolated

Isolated

Observed System

Disruptive Workload

W_{d_{\text{end}}}

W_{d_{\text{base}}}

W_{d_{\text{ref}}}

Performance Isolation
Metrics Based on Workload Ratio Integrals

- Abiding workload
- Non-Isoalted
- Isolated
- Observed System

\[ W_{a_{\text{ref}}} \]
\[ W_{a_{\text{base}}} \]
\[ W_{d_{\text{ref}}} \]
\[ W_{d_{\text{base}}} \]
\[ W_{d_{\text{end}}} \]
\[ p_{\text{end}} \]

Disruptive workload

48  S. Kounev  Resource Elasticity  Performance Isolation  Conclusions
Metrics Based on Workload Ratio Integrals

\[ I = \frac{A_{\text{measured}} - A_{\text{nonIsolated}}}{A_{\text{isolated}} - A_{\text{nonIsolated}}} \]
Example Metrics: $I_{\text{intBase}}$ and $I_{\text{intFree}}$

\[ I_{\text{intBase}} = \frac{\left( \frac{W_{\text{d_base}}}{W_{\text{d_ref}}} \int_{W_{\text{d_ref}}}^{W_{\text{d_base}}} f_m(W_d) \, dW_d \right) - W_{a_{\text{ref}}}^2 / 2}{W_{a_{\text{ref}}}^2 / 2} \]

\[ I_{\text{intFree}} = \frac{\left( \frac{p_{\text{end}}}{W_{\text{d_ref}}} \int_{W_{\text{d_ref}}}^{p_{\text{end}}} f_m(W_d) \, dW_d \right) - W_{a_{\text{ref}}}^2 / 2}{W_{a_{\text{ref}}} \cdot (p_{\text{end}} - W_{\text{d_ref}}) - W_{a_{\text{ref}}}^2 / 2} \]

Areas within $W_{\text{d_ref}}$ and $W_{\text{d_base}}$

Areas within $W_{\text{d_ref}}$ and predefined bound.

Perfectly Isolated = 1

Non-Isolated = 0

Answers: How much potential has the isolation method to improve?
Case Study

Add Delay  Round Robin  Blacklist  Separate Thread Pools

Three Components of Reliable Benchmarking

Reliable Metrics
- What exactly should be measured and computed?

Representative Workloads
- For which scenarios and under which conditions?

Sound Measurement Methodology
- How should measurements be conducted?

“To measure is to know.” -- Clerk Maxwell, 1831-1879

“It is much easier to make measurements than to know exactly what you are measuring.” -- J.W.N.Sullivan (1928)
Conclusion

- Use of individual metrics in isolation can provide misleading impression
- To understand the overall system behavior, we need multiple metrics reflecting different aspects
- We also need representative workloads and a sound measurement methodology
- **Open-Systems-Group (OSG)**
  - Processor and computer architectures
  - Virtualization platforms
  - Java (JVM, Java EE)
  - Message-based systems
  - Storage systems (SFS)
  - Web-, email- and file server
  - SIP server (VoIP)
  - Cloud computing

- **High-Performance-Group (HPG)**
  - Symmetric multiprocessor systems
  - Workstation clusters
  - Parallel and distributed systems
  - Vector (parallel) supercomputers

- **“Graphics and Workstation Performance Group” (GWPG)**
  - CAD/CAM, visualization
  - OpenGL
SPEC Research Group (RG)

- Founded in March 2011
  - Transfer of knowledge btw. academia and industry

- Activities
  - Methods and techniques for experimental system analysis
  - Standard metrics and measurement methodologies
  - Benchmarking and certification
  - Evaluation of academic research results

- Member organizations (Feb 2014)
Thank You!

skounev@acm.org
http://se.informatik.uni-wuerzburg.de