A Cloud Benchmark Suite Combining Micro and Application Benchmarks

Joel Scheuner, Philipp Leitner

Joel Scheuner

✉️ scheuner@chalmers.se
👋 joe4dev
🐦 @joe4dev
Context: Public Infrastructure-as-a-Service Clouds

Infrastructure-as-a-Service (IaaS)
Platform-as-a-Service (PaaS)
Software-as-a-Service (SaaS)
Motivation: Capacity Planning in IaaS Clouds

What cloud provider should I choose?

https://www.cloudorado.com
Motivation: Capacity Planning in IaaS Clouds

What cloud service (i.e., instance type) should I choose?

- **t2.nano**
  - 0.05-1 vCPU
  - 0.5 GB RAM
  - $0.006/h

- **x1e.32xlarge**
  - 128 vCPUs
  - 3904 GB RAM
  - $26.688 hourly
Topic: Performance Benchmarking in the Cloud

“The instance type itself is a very major tunable parameter”

@brendangregg re:Invent’17
https://youtu.be/89fYOo1V2pA?t=5m4s
Background

**Micro Benchmarks**
- CPU
- Memory
- I/O
- Network

**Application Benchmarks**
- Overall performance (e.g., response time)

**Generic**
- Artificial

**Specific**
- Real-World

**Resource-specific**
- Resource-heterogeneous
Related Work

Micro Benchmarking / Application Kernels
Iosup et. al. Performance analysis of cloud computing services for many-tasks scientific computing. Ostermann et. al. A performance analysis of EC2 cloud computing services for scientific computing.

Application Benchmarking
Ferdman et. al. Clearing the clouds: a study of emerging scale-out workloads on modern hardware. Cooper et. al. Benchmarking Cloud Serving Systems with YCSB.

Repeatability of Cloud Experiments
Abedi and Brecht. Conducting Repeatable Experiments in Highly Variable Cloud Computing Environments. @ICPE’17
Problem: Isolation, Reproducibility of Execution

Micro Benchmarks
- CPU
- Memory
- I/O
- Network

Generic
Artificial
Resource-specific

Application Benchmarks
Overall performance (e.g., response time)

Specific
Real-World
Resource-heterogeneous
Question: How can we systematically combine and execute micro and application benchmarks?

Micro Benchmarks
- CPU
- Memory
- I/O
- Network

Application Benchmarks
- Overall performance (e.g., response time)

Generic
- Artificial
- Resource-specific

Specific
- Real-World
- Resource-heterogeneous
Idea

Micro Benchmarks
- CPU
- Memory
- I/O
- Network

Application Benchmarks
- Overall performance (e.g., response time)

Generic
Artificial
Resource-specific

Systematically Execute Together

Specific
Real-World
Resource-heterogeneous
Execution Methodology

**Execution Methodology**

- **Multiple Consecutive Trials**: All trials for the first alternative are conducted, followed by the second alternative and each of the remaining alternatives. Figure 1-B shows the Multiple Consecutive Trials technique for 3 alternatives.

- **Randomized Multiple Interleaved Trials (RMIT)**: In this approach, an experiment consists of only a single trial. Figure 1-A shows an example of this approach with 3 trials. D) Randomized Multiple Interleaved Trials (RMIT)

- **RMIT**: 30 benchmark scenarios, 3 trials, ~2-3h runtime

---

**Figure 1**

- **Acquire Resources**
- **Provision VM**
- **Start Benchmarks**
- **Benchmark Results**
- **Release Resources**
- **Run Benchmark**

**Cloud Instance**

**Provider API**

**Benchmark Manager**

**Amazon EC2**
Benchmark Manager

Cloud WorkBench (CWB)
Tool for scheduling cloud experiments

sealuzh/cloud-workbench

Scheuner, Leitner, Cito, and Gall

Demo@WWW 2015
Scheuner, Cito, Leitner, and Gall
Architecture Overview
Micro Benchmarks

Broad resource coverage and specific resource testing

**CPU**
- sysbench/cpu-single-thread
- sysbench/cpu-multi-thread
- stressng/cpu-callback
- stressng/cpu-double
- stressng/cpu-euler
- stressng/cpu-ftt
- stressng/cpu-fibonacci
- stressng/cpu-int64
- stressng/cpu-loop
- stressng/cpu-matrixprod

**Memory**
- sysbench/memory-4k-block-size
- sysbench/memory-1m-block-size

**I/O**
- [file I/O] sysbench/fileio-1m-seq-write
- [file I/O] sysbench/fileio-4k-rand-read
- [disk I/O] fio/4k-seq-write
- [disk I/O] fio/8k-rand-read

**Network**
- iperf/single-thread-bandwidth
- iperf/multi-thread-bandwidth
- stressng/network-epoll
- stressng/network-icmp
- stressng/network-sockfd
- stressng/network-udp

**Software (OS)**
- sysbench/mutex
- sysbench/thread-lock-1
- sysbench/thread-lock-128
Micro Benchmarks: Examples

File I/O: 4k random read

1) Prepare
2) Run
3) Extract Result
4) Cleanup

File I/O: 4k random read

Result
972 Mbits/sec

Network
Bandwidth

CPU
Memory
I/O
Network

3.5793 MiB/sec

Server
Client

Result
972 Mbits/sec
Application Benchmarks

Molecular Dynamics Simulation (MDSim)

WordPress Benchmark (WPBench)

Multiple short blogging session scenarios (read, search, comment)
## Performance Data Set

<table>
<thead>
<tr>
<th>Instance Type</th>
<th>vCPU</th>
<th>ECU*</th>
<th>RAM [GiB]</th>
<th>Virtualization</th>
<th>Network Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>m1.small</td>
<td>1</td>
<td>1</td>
<td>1.7</td>
<td>PV</td>
<td>Low</td>
</tr>
<tr>
<td>m1.medium</td>
<td>1</td>
<td>2</td>
<td>3.75</td>
<td>PV</td>
<td>Moderate</td>
</tr>
<tr>
<td>m3.medium</td>
<td>1</td>
<td>3</td>
<td>3.75</td>
<td>PV /HVM</td>
<td>Moderate</td>
</tr>
<tr>
<td>m1.large</td>
<td>2</td>
<td>4</td>
<td>7.5</td>
<td>PV</td>
<td>Moderate</td>
</tr>
<tr>
<td>m3.large</td>
<td>2</td>
<td>6.5</td>
<td>7.5</td>
<td>HVM</td>
<td>Moderate</td>
</tr>
<tr>
<td>m4.large</td>
<td>2</td>
<td>6.5</td>
<td>8.0</td>
<td>HVM</td>
<td>Moderate</td>
</tr>
<tr>
<td>c3.large</td>
<td>2</td>
<td>7</td>
<td>3.75</td>
<td>HVM</td>
<td>Moderate</td>
</tr>
<tr>
<td>c4.large</td>
<td>2</td>
<td>8</td>
<td>3.75</td>
<td>HVM</td>
<td>Moderate</td>
</tr>
<tr>
<td>c3.xlarge</td>
<td>4</td>
<td>14</td>
<td>7.5</td>
<td>HVM</td>
<td>Moderate</td>
</tr>
<tr>
<td>c4.xlarge</td>
<td>4</td>
<td>16</td>
<td>7.5</td>
<td>HVM</td>
<td>High</td>
</tr>
<tr>
<td>c1.xlarge</td>
<td>8</td>
<td>20</td>
<td>7</td>
<td>PV</td>
<td>High</td>
</tr>
</tbody>
</table>

* ECU := Elastic Compute Unit (i.e., Amazon’s metric for CPU performance)

>240 Virtual Machines (VMs) à 3 Iterations → ~750 VM hours

>60'000 Measurements (258 per instance)
WPBench Response Time Cost Frontier

Cost/Performance is a trade-off but there exist unfavorable instance types

-80% performance -35% cost
+40% performance - 40% cost
Almost perfect stability in comparison to previous results

The newer virtualization type hvm is more I/O efficient than pv.
Future Work

Benchmark Design
Benchmark Execution
Data Pre-Processing
Data Analysis

QUDOS@ICPE 2018 “A Cloud Benchmark Suite Combining Micro and Applications Benchmarks”
Scheuner and Leitner

“Estimating Cloud Application Performance Based on Micro Benchmark Profiling”
Scheuner and Leitner
Conclusions

Selecting an optimal instance type can save up to 40% costs while increasing up to 40% performance

Support trend towards more predictable performance (AWS EC2)

The newer virtualization type (hvm) improves I/O utilization rates up to 10% (vs pv)

scheuner@chalmers.se