Seminar – Advanced topics on Information Systems
Benchmarking the Cloud

Jérémie Miserez
December 11th, 2010
Overview

- Introduction
- Cloud services
- Benchmarking a cloud service
  - Metrics
  - Requirements for applications
- Running the TPC-W benchmark
  - Restrictions and adaptations
  - Implementations
  - Results
- Future benchmarks
Focus

- Running an application *with a database* in the cloud.
- Using *public* cloud service providers
Cloud services

Why should I use a cloud service? Which one?

- Essentially a business decision

- Need to:
  - Understand differences and limitations
  - Compare different offerings
Cloud services (2)

<table>
<thead>
<tr>
<th>Traditional system</th>
<th>System in the cloud</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Fixed total cost, known in advance</td>
<td>• Variable total cost</td>
</tr>
<tr>
<td>• Peak performance requirements determine total hardware cost.</td>
<td>• Actual usage determines total cost.</td>
</tr>
<tr>
<td>• High utilization (e.g. batch processing) lowers the cost per data set.</td>
<td>• Ideally, the workload has little effect on cost per data set.</td>
</tr>
<tr>
<td>• High throughput and low latency lower the cost per data set.</td>
<td>• Performance has little effect on cost per data set.</td>
</tr>
<tr>
<td>• Maximize performance for a given hardware configuration.</td>
<td>• Minimize cost for a given set of requirements (performance, consistency, scalability, etc.)</td>
</tr>
</tbody>
</table>
Cloud services (3)
Cloud services (4)

IaaS & PaaS
- Elastic Compute Cloud (EC2)
- Elastic Block Storage (EBS)
- Simple Storage Service (S3)
- Relational Database Service (RDS)
- SimpleDB Service
- ...

IaaS & PaaS
- Azure Compute
- Azure Storage:
  - BLOB service
  - Table service
  - Queue service
- SQL Azure

PaaS
- AppEngine
- Java and Python runtime environments
- Distributed data storage service
- Supports embedded GQL (Google Query Language)
Benchmarkeding cloud services

What to measure (metrics)?

- Traditionally:
  - Need to maximize performance for a given hardware configuration.
  - Measure performance (throughput, latency)

- Cloud based:
  - Need to minimize cost for a given set of requirements.
  - Measure cost, cost predictability, scalability
    - Need to measure performance as a function of system size!
  - Possibly separate measurements for different consistency levels
Benchmarking cloud services (2)

Requirements?

- Minimal acceptable throughput, maximal acceptable response time
- Aggressiveness of scaling, what delays are acceptable?
- Consistency: what level of consistency is needed, maximal acceptable propagation time for eventual consistency
Benchmarking cloud services (3)

How to measure?

- Different requirements may call for different implementations.
- Implementations may behave differently depending on the cloud service provider.
Running the TPC-W benchmark

Simulate a simple electronic book store.

- Emulated browsers (EBs) browse the store and buy books.
- Each EB waits for a response before it sends the next request.

- TPC-W obsolete, but still popular.
Running the TPC-W benchmark (2)

The original TPC-W metrics are as follows:

- Web interactions per second (WIPS): The amount of valid requests per second.
  - A request is valid if a correct response is received during the allowed response time
- Cost/WIPS: The TPC-W benchmark defines rules on how to calculate this metric.
Running the TPC-W benchmark (3)

New metrics and adaptations for the cloud:

- Cost directly available
- WIPS(EB) instead of WIPS
- Cost/WIPS(EB) instead of Cost/WIPS
- $s(\text{Cost/WIPS})$: Standard deviation, cost predictability
- Database size remains fixed for all tests (10000 items)
- Consistency: The TPC-W benchmark requires strong consistency. Not every implementation guarantees this.
Running the TPC-W benchmark (4)

- **Classic**: HTTP, Web Server + App Server, DB Server, Storage.
- **Partitioning**: HTTP, Web Server + App Server, DB Server, Storage.
- **Distributed (+Cache)**: HTTP, App Server + Web Server + DB Server, Clients, Storage.
## Running the TPC-W benchmark (5)

Total of 7 implementations built:

<table>
<thead>
<tr>
<th></th>
<th>AWS MySQL</th>
<th>AWS MySQL/R</th>
<th>AWS RDS</th>
<th>AWS SimpleDB</th>
<th>AWS S3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Architecture type</strong></td>
<td>Classic</td>
<td>Replication</td>
<td>Classic</td>
<td>Partitioning, Replication</td>
<td>Distributed Control</td>
</tr>
<tr>
<td><strong>Database type</strong></td>
<td>MySQL</td>
<td>MySQL Rep</td>
<td>MySQL</td>
<td>SimpleDB</td>
<td>-</td>
</tr>
<tr>
<td><strong>Query language</strong></td>
<td>SQL</td>
<td>SQL</td>
<td>SQL</td>
<td>SimpleDB</td>
<td>-</td>
</tr>
<tr>
<td><strong>Reliable Storage</strong></td>
<td>EBS</td>
<td>EBS, EC2 (slaves)</td>
<td>-</td>
<td>-</td>
<td>S3</td>
</tr>
<tr>
<td><strong>Consistency</strong></td>
<td>Repeatable read</td>
<td>Repeatable read</td>
<td>Repeatable read</td>
<td>Eventual consistency</td>
<td>Eventual consistency</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Google App Engine</th>
<th>Microsoft Azure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Architecture type</strong></td>
<td>Partitioning, Replication, Caching</td>
<td>Replication</td>
</tr>
<tr>
<td><strong>Hardware conf.</strong></td>
<td>Automatic</td>
<td>Manual/Automatic</td>
</tr>
<tr>
<td><strong>Database type</strong></td>
<td>DataStore</td>
<td>SQL Azure</td>
</tr>
<tr>
<td><strong>Query language</strong></td>
<td>GQL</td>
<td>SQL</td>
</tr>
<tr>
<td><strong>Reliable Storage</strong></td>
<td>Google Filesystem (GFS)</td>
<td>Windows Azure</td>
</tr>
<tr>
<td><strong>Consistency</strong></td>
<td>Snapshot Isolation</td>
<td>Snapshot Isolation</td>
</tr>
</tbody>
</table>
Running the TPC-W benchmark (6)

Results - Performance:
Running the TPC-W benchmark (7)

Results:

Cost per WIPS
1 - 1000 EB (lin-log scale)
Running the TPC-W benchmark (8)

Results:

Cost per WIPS
10 - 1000 EB (linear scale)
Running the TPC-W benchmark (9)

Results:

![Graph showing total cost per day for various services including MySQL, MySQL/R, RDS, SimpleDB, S3, Google AE, Google AE/C, and Azure. The graph plots total cost per day against EB (Equivalent Bandwidth) ranging from 1 to 9000 EB.]
Running the TPC-W benchmark (10)

Results – Cost composition (250 EBs):
Running the TPC-W benchmark (11)

Results - Scalability:

- **Scale-out:**
  - In all variants, additional web and applications servers were added in order to sustain the workload.
  - However, only with the Amazon S3 implementation is it possible to add an arbitrary amount of database servers to the cloud.

- **Scale-up:**
  - Amazon RDS offers the option to scale up to a larger database server at any time. This is the only way for the MySQL variants to improve throughput.
  - Microsoft Azure offers no option to scale up, but seems to provide a powerful enough database server automatically.
Outlook

More to test:

- **Flexibility**: How fast can the service adapt to peak workloads?
- **Availability and fault tolerance**: How likely is failure of an instance?
- **How does performance change with different consistency guarantees?**
Outlook (2)

Is the TPC-W benchmark an appropriate tool?

- Inner workings of public clouds are not generally published
- Therefore, a benchmark should be selected that approximates the real application to be deployed.
  - The modified TPC-W benchmark fulfills this criteria.
- Other benchmarks specifically designed for cloud services are actively being developed.
  - Difficult to compare raw performance numbers when every cloud service behaves differently or does not support some features.
Comments / Questions?