DISTRIBUTED RESEARCH ON EMERGING APPLICATIONS & MACHINES
dream-lab.in | **Indian Institute of Science**, Bangalore

SE252:Lecture 21, Mar 31

ILO5: Performance & Consistency *(Performance Benchmarks & Monitoring)*

Yogesh Simmhan

DREAM:Lab

IL05:Performance & Consistency

ILO 5: Performance & Consistency on Clouds

- *Describe* and *compare* different performance metrics for evaluating Cloud applications and
- *demonstrate* their use for application measurement.

➲Project

- *Explain* the distinctions between Consistency, Availability & Partitioning (CAP theorem)
- *Discuss* the types of Cloud applications that exhibit these features.

Evaluating an Application

- How well does the application perform?
  - On one or more types of systems
    - » E.g. S/M/L VMs
  - Different inputs, datasets, users, …
    - » M/L/XL files
  - Using one or more configurations
    - » Dataflow mapping to resources
- "perform well"
  - Property: Performance
  - Metrics: Makespan, Latency

Application Benchmarking

Evaluating a System

- How well does the system perform?
  - **System under test (SUT):** Average Performance under a Particular Workload
  - On different types of workloads
    - » Collection of applications
    - » IO/compute intensive, long/short
  - Under different system conditions
    - » Temporal/Spatial Variability
- "perform well"
  - Property: Scalability (Weak/Strong)
  - Metrics: Speedup, Throughput

System Benchmarking

Why Benchmark?

- System selection for an application
  - *What is the expected performance of my VM under different conditions?*
- Service Level Agreement that can be given
  - *What can I charge for my VM? What can I guarantee to my users?*
- Uniform comparison across systems
  - *Which is the better system among two or more?*
- Micro-benchmarks **vs.** Application Workload Benchmarks
  - Test individual components vs.
  - Test system stack for application classes

How to Build a Benchmark

Kistowski, et al, ICPE'15

- **Relevance:** How closely the benchmark behavior correlates to behaviors that are of interest to consumers of the results
- **Reproducibility:** The ability to consistently produce similar results when the benchmark is run with the same test configuration
- Fairness: Allowing different test configurations to compete on their merits without artificial limitations
- **Verifiability:** Providing confidence that a benchmark result is accurate
- **Usability:** Avoiding roadblocks for users to run the benchmark in their test environments

Benchmarking in the Cloud†

- Building & Running a Benchmark for the Cloud
- Building a Benchmark
  - Workload Parameters
  - System Configuration
  - Metrics to Measure
- Running a Benchmark
  - Artifacts, outliers, repeatability, …

† Benchmarking in the Cloud: What it Should, Can, and Cannot Be, Enno Folkerts, et al., *TPCTC*, 2012

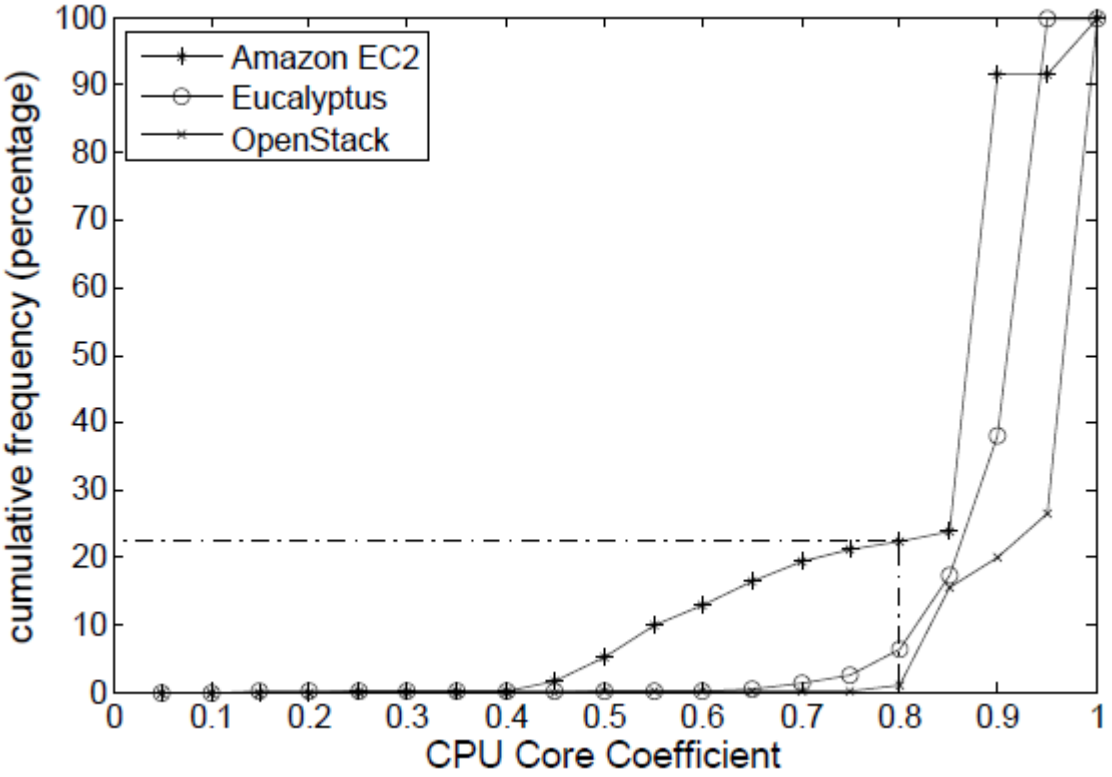
Meaningful Metrics

- *Ability to interpret a measured result*
  - Individual App: Makespan
  - Collection of Apps *(maxInst)*: Throughput
- Price vs. Performance Metric
  - *"Minimal cost to run maxInst of a given application and a given workload on the SUT"*
  - Costs for single and collections of machines, when scaling

Meaningful Metrics

- Elasticity Metric
  - Over & Under Provisioning of Resources
  - SLA violations under elasticity
  - "Normal" variations vs. Elasticity variations
  - *Does a variable/elastic workload lead to a lower price?*
- Percentile
  - Measure the metric at the 90/95/99th Percentile
  - Representative, Eliminate outliers
  - Cumulative Distribution Function
    - » X axis is app # sorted by fastest to slowest
    - » Y axis is completion time, relative to start

CDF


Amazon EC2
Eucalyptus
OpenStack
cumulative frequency (percentage)
CPU Core Coefficient
0
10
20
30
40
50
60
70
80
90
100
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Workload Design

- Model real-world design
- Limit the Resources Acquired
  - "Infinite resources"
  - Maximum scaling factor
    - » Do not scale till you break
- Variable Load
  - E.g. Elasticity, adaptivity
  - Slow changing, harmonic, random walk
  - Capture realistic application pattern

Workload Design

- Scalability
  - Measure metric as some factor increases
  - ↑ number of users per application instance.
  - ↑ size of the application instance.
  - ↑ number of application instances.
  - ↑ number of application instance per node.

Workload Implementation

- Workload Generation
  - Synthetic versions of applications with distribution of characteristics, random datasets
  - Complexity of reference application (e.g. dataflow, MapReduce)
  - Ensure scaling is not affected
  - Capture relevant characteristics

Workload Implementation

- Fairness and Portability
  - Reproducibility & Runnable across systems
  - Restrictions on implementation, technology may be unfair

1. Set application specs without touching implementation details.
2. Service providers are free to provide their own optimal implementation of the respective application.
3. Service providers can require submissions to be based on their own implementation.

Final Project Report & Evaluation

- *Report should include the following topics. Final Demo should highlight these. Final code on GitHub should have README instructions.*
- Cloud Virtualization and Enabling Technologies (ILO2) [25%]
  - » Introduction of application/project
  - » Virtualization/Web Service/Storage and other Cloud/Big Data technology used
  - » Motivation & benefits of using above technologies for your application
- Programming Pattern on Clouds (ILO3) [25%]
  - » DAG/pattern used by your application
  - » Degrees of parallelization, data locality
  - » Potential speedup, weak/strong scaling
- Scheduling & Scalability (ILO4) [25%]
  - » Scheduling & load balancing models that can/have been used to scale
  - » Possible bottlenecks & how you could avoid it
  - » Benefits of elastic scaling in/out on Clouds
- Performance Results (ILO5) [25%]
  - » Benchmark configuration for evaluating your app
  - » Metrics for success & justification on why these matter
  - » Results of running app benchmark on Cloud VMs
  - » Analysis of expected vs. observed behaviour.

Assignments

- **Homework B** posted. Due on Fri 10 Apr by midnight.
- Lookup an *industry standard benchmark* and discuss at next class (e.g. TPC, SPEC)