EE 542, Lectures 14 ~ 15:
October 9 and 11, 2017

Cloud Performance and Scaling Techniques

Readings: Sections 10.1, 10.2 and 10.3 in Chapter 10

Prof. Kai Hwang, USC

.

Cloud Performance Issues:

1) *Scaling measurement:* Cloud scaling is done with virtualized resources. Hence, the scale of computing power is decided at various abstraction levels of virtual resources.

2) *Workload scenario:* Cloud aims to accommodate workload with large number of small jobs. Scaling strategies must match with such a workload scenario.

Cloud Performance Issues:

3) *Performance attributes:* To benefit a large number of small jobs, performance concerns are the response time and throughput, rather than batch execution time.

4) *Cloud productivity:* Productivity is tied to performance cost ratio. Tradeoffs do exist in high performance versus service costs to massive users.

Scale-Out Workloads

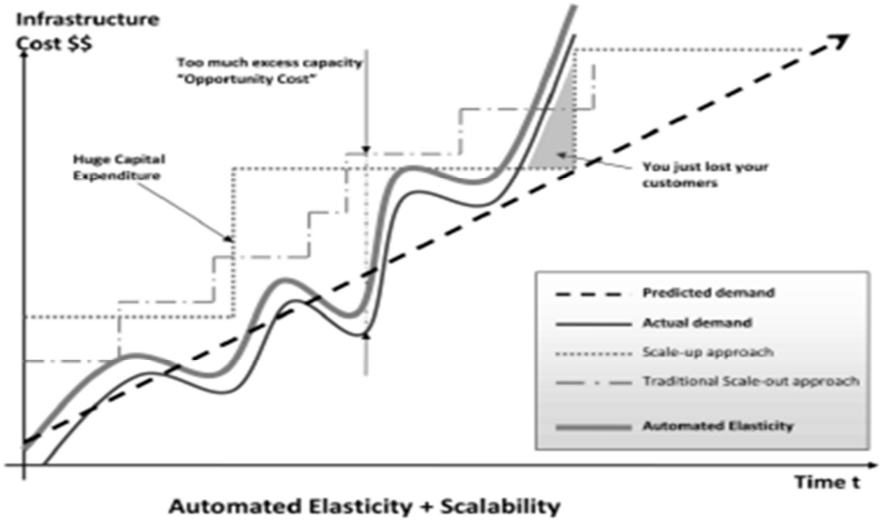
Cloud workloads are characterized by their dataset size, algorithms, memory-access pattern, and service model applied. Scaling techniques cover three cloud workload types :

- *Scale-out technique* allows adding more machine instances or processing nodes of the same type based on the quota agreed in the *service-level agreement* (SLA). Obviously, scaling out appeals more to the use of homogeneous clusters with identical nodes.

Scale-Up and Mixed Workloads

- ***Scale-up technique*** **is implemented with scaling the cloud from using small nodes to more powerful nodes equipped with better processor, memory or storage.**
- ***Mixed scale-up/scale-out technique*** **allows one to scale up or scale-down the instance type and adjust the instance quantity by scale-out (increasing) or scale-in (reducing) resources at the same time. Mixed scaling appeals better with using heterogeneous clusters.**

Understanding Elasticity in Cloud Resources



Elastic Cloud Resources Provisioning for High Throughput Performance

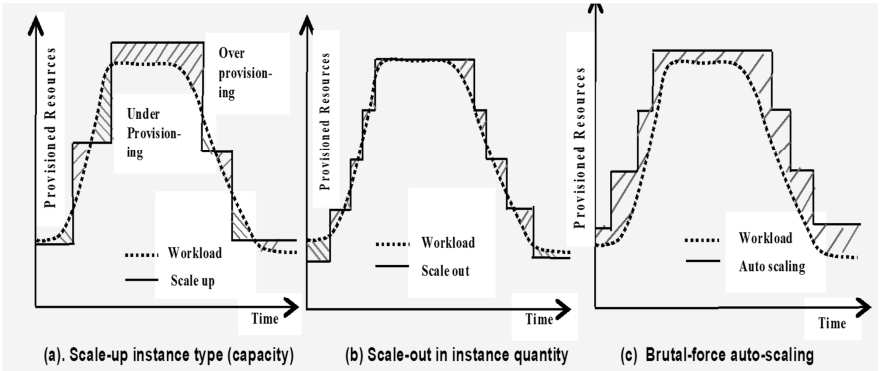


Figure 1: Auto-scaling, scale-out and scale-up machine instance resources in elastic IaaS clouds, where over-provisioning and under-provisioning of machine resources are shown in differently shaded areas above and below the workload curves. .

TABLE 2. PERFORMANCE, CAPABILITY AND PRODUCTIVITY METRICS FOR EVALUATING CLOUDS

Abstraction Level	Performance Metric	Notation (Eq. #)	Brief Definitions with Representative Units or Probabilities
Basic Performance Metrics	*Execution time*	T_e	Time elapsed during program or job execution, (sec., hours)
	Speed	S_r	Number of operations executed per second, (PFlops, TPS, WIPS, etc.)
	Speedup	S_u	Speed gain of using more processing nodes over a single node
	Efficiency	E	Percentage of max. Performance (speedup or utilization) achievable (%)
	Scalability	S (Eq.5)	The ability to scale up resources for gain in system performance
	Elasticity	E_l (Eq.14)	Dynamic interval of auto-scaling resources with workload variation
Cloud Capabilities:	*Latency*	T	Waiting time from job submission to receiving the first response. (Sec.)
	Throughput	H	Average number of jobs/tasks/operations per unit time (PFops, WIPS.)
	Bandwidth	B	Data transfer rate or I/O processing speed, (MB/s, Gbps)
	Storage Capacity	S_g	Storage capacity with virtual disks to serve many user groups
	Software Tooling	S_w	Software portability and API and SDK tools for developing cloud apps.
	Bigdata Analytics	A_n	The ability to uncover hidden information and predict the future
	Recoverability	R_c	Recovery rate or the capability to recover from failure or disaster (%)
Cloud Productivity	*QoS of Cloud*	*QoS*	The satisfaction rate of a cloud service or benchmark testing (%)
	Power Demand	W	Power consumption of a cloud computing system (MWatt)
	Service cost	*Cost*	The price per cloud service (compute, storage, etc.) provided, ($/hour)
	SLA/Security	L	Compliance of SLA , security, privacy or copyright regulations
	Availability	A	Percentage of time the system is up to deliver useful work. (%)
	Productivity	P, (Eq.4)	Cloud service performance per unit cost, (TFlops/$, WIPS/$, etc.)

Basic Performance Metrics :

Speed **(*S*)** ***:*** **Number of *millions of operations per second* (*Mops*). The operation could be integer or floating-point like *MFlops.* The speed is also called *throughput* such as *millions of web interactions per second* (*MIPS*), etc.**

Speedup **(S_u): Speed gain of using multiple nodes**

Efficiency **(E_f): Percentage of peak performance achieved**

Utilization **(*U*)*:* Busy resources (CPU, memory, storage).**

Scalability **(*S*) *:* Scaling ability to upgrade performance.**

Cloud Capabilities are macroscopic metrics :

Latency **(*L*): System response time or access latency**

Bandwidth **(*B*): This is data transfer rate or I/O rate.**

Elasticity **(E_l): The ability for cloud resources to scale up/down or scale in/out to match with workload variation**

Software **(S_w) : Software portability, API and SDK tooling**

Big-data Analytics: **(A_n)*:* The ability to uncover hidden information or predict trends in big data.**

.Cloud *Productivity Measures*:

Quality of Service **(*QoS*): Satisfaction on user services**

System availability **(*A*)*: The s*ystem up time per year.**

Service costs **(C_o): User renting costs and provider cost.**

Power Demand **(*W*)*:* Cloud power consumption (MWatt).**

SLA/Security **(*L*) *:* Compliance of SLA, security, etc.**

Productivity **(*P*) *:* QoS-satisfied perf. per unit cost**

Elasticity Analysis of Scalable Cloud Performance

- **Elasticity refers to the stretchability of cloud resources like CPU power, storage capacity, etc.**
- **Elasticity is measured by the speed (or overhead) to reconfigure the cloud configuration.**
- **Elastic resource provisioning cannot not be done with physical machine, only virtual machines can be refigured in real-time.**

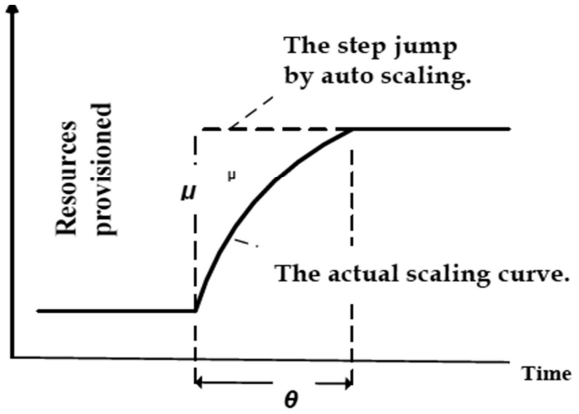


Figure 3. Illustration of cloud resource provisioning, where θ is the overhead time and μ is the offset between actual scaling and an auto scaling process.

Cloud Elasticity Analysis (cont'd)

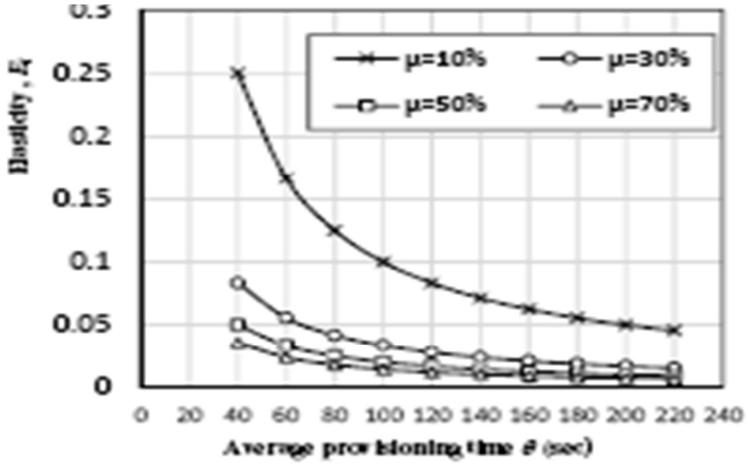


Figure 4. The cloud elasticity plotted from Eq.21.

Elastic Compute Unit (ECU)

- **Amazon AWS has defined a term : *Elastic (or EC2) Compute Unit* (ECU) as an abstract unit to quantify the computing capacity of a machine instance type.**
- **By 2009 standard, the performance of a 1 ECU instance is roughly equivalent to the CPU capacity of a 1.2 GHz 2007 Xeon processor.**
- **Each physical CPU (processor) can house a number a number of *virtual CPU* (vCPU). Also the memory and storage may affect the ECU count.**

Performance Metrics: Speedup and Efficiency

Given a problem size x and n-processors system, the *speedup*(n,x) is the sequential execution time $Time(1,x)$ divided by parallel execution time $Time(n,x)$.

$$Speedup(n,x) = (Time(1,x))/(Time(n,x)) \quad (1)$$

The *efficiency*(n,x) is defined by the following ratio :

$$Efficiency(n,x) = \frac{Speedup(n,x)}{n} = (Time(1,x))/(n \times Time(n,x)) \quad (2)$$

Cloud Efficiency for Different EC2 Configurations

Consider a cluster configuration A. Let $T(1)$ be the execution time of an application code on a 1-ECU instance. Let $T(A)$ be the execution time of the same code on a virtual cluster A. The speedup is defined by $Speedup\ (A) = T(1) / T(A)$. Assume that the cluster is built with n instance types. The type-i has n_i instances, each with an ECU count c_i. We calculate the total cluster ECU count by:

$$N(A) = \sum_{i=1}^{i=n} n_i \times c_i \qquad (5)$$

Table 10.3
The ECU rating of machine instance types in Amazon EC2 in 2014

Instance Type	ECU	Virtual Cores	Memory (GB)	Storage (GB)	Price ($/hour)
m1.small	1	1	1.7	1×160	0.044
m1.medium	2	1	3.7	1×410	0.087
m3.medium	3	1	3.75	1×4 SSD	0.07
m1.xlarge	8	4	15	4×420	0.350
m3.xlarge	13	4	15	2×40 (SSD)	0.280
c1.xlarge	20	8	7	4×420 (SSD)	0.520
c3.xlarge	14	4	7.5	2×40 (SSD)	0.210

Cloud Efficiency for Diffent EC2 Configurations

This $N(A)$ count sets a ceiling of the cluster speedup. Now, we are ready to define the *cloud efficiency* for the cluster A in question as follow:

$$Efficiency\,(A) = Speedup\,(A) / N(A)$$

$$= T(1) / \{T(A) \times \sum_{i=1}^{i=n} n_i \times c_i\} \qquad (6)$$

Cloud Productivity

In general, the cloud *productivity* is driven by three technical factors that are related to the scaling factor.

1) System performance such as throughput in terms of transactions per second or response time.

2) System availability as an indicator of QoS measured by percentage of uptime.

Cloud Productivity

3) Cost for rented resources measured by price.

Let Λ be a cloud configuration in use. We define the cloud *productivity* by three factors, all are functions of Λ.

$$P(\Lambda) = \frac{p(\Lambda)\times\omega(\Lambda)}{C(\Lambda)} \quad (7)$$

where $p(\Lambda)$ is a *performance* metric used, which could the speed or throughput selected from the last section. The $\omega(\Lambda)$ is the *QoS* of the cloud. For simplicity, one can approximate the QoS by the *service availability* measure. According to CloudHarmonics Report on 144 cloud web sites [5], more than half have 99% or higher availability. The $C(\Lambda)$ is the user cost to rent resources to form the virtual cluster Λ.

Cloud Scalability

The *cloud scalability* is driven by the productivity and QoS of a cloud system. This measure is inversely proportional to the service costs As we scale from configuration $\Lambda 1$ to another $\Lambda 2$. This metric evaluates the economy of scale by a pair of productivity ratio. The higher is the value of a scalability measure, the more opportunity exists to target the desired scaling scheme.

Cloud Scalability

$$\Phi(\Lambda 1, \Lambda 2) = \frac{P(\Lambda 2)}{P(\Lambda 1)} = \frac{p(\Lambda 2)\times\omega(\Lambda 2)\times C(\Lambda 1))}{p(\Lambda 1)\times\omega(\Lambda 1)\times C(\Lambda 2)} \quad (8)$$

With comparable QoS and cost estimation, the scalability is directly proportional to productivity (Eq.7). Therefore, will demonstrate the measured productivity results and skip the scalability plots in subsequent sections.

A Generic Cloud Performance Model

$$F(\textit{cloud model}) = \{ < \textit{service offerings} >, < \textit{benchmark or app code under test} >, < \textit{performance metrics} >, < \textit{performance map generated} > \} \quad (9)$$

where the *cloud model* could be one or more of the available service modes such as IaaS (*Infrastructure as a Service*), PaaS (*Platform as a Service*) , SaaS (*Software as a Service*), HaaS (*Health-care as a Service*), etc. [12].

Lecture 15: Oct. 11, 2017
Cloud Performance Modeling

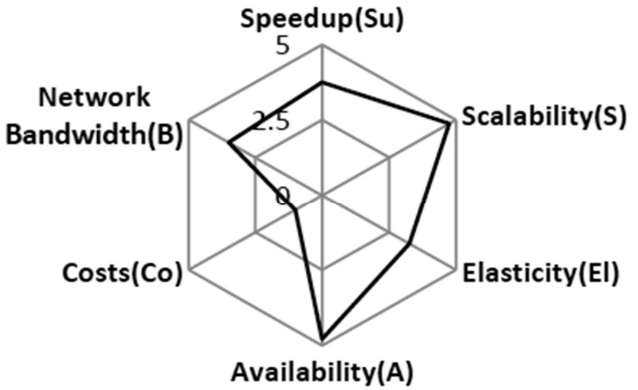
A. Modeling IaaS Cloud Performance

The IaaS model assumes a flat architecture providing infrastructure to client users. The IaaS providers are concerned about resource utilization and power consumptions. The end users are concerned about response time and prices, etc. The PaaS and SaaS vendors may apply a hierarchical architecture. This graphic model can be used to evaluate many other IaaS clouds such as Rackspace, GoGrid, FlexScale, etc. [12].

$$F(IaaS) = \{ < EC2, S3, \ldots >, < Benchmarks >, <S_u, E_l, \Phi, B, A, C_o >, < Perf.\ Map > \} \quad (10)$$

IaaS Cloud Performance

$$F\ (Infrastructure\ cloud) = \{< IaaS >, < Compute, Storage >, <S_u, E_l, S >, < B >, < A, C_o > \}$$



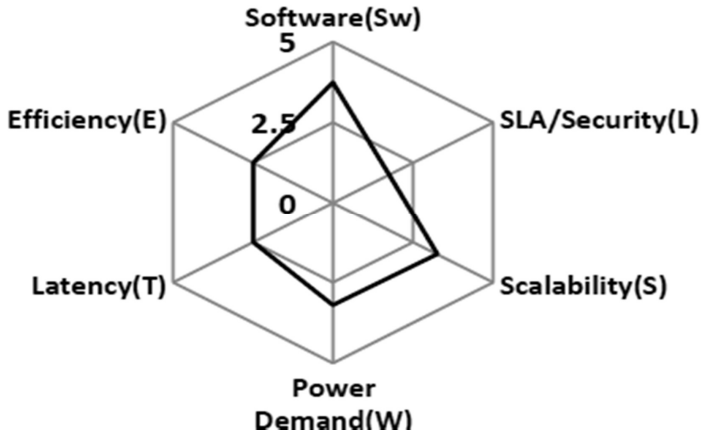
Each cloud performance model is specified by a 4-tuple expression, where *f* is the model function.

$$F\ (Cloud\ model) = \{ < Service\ offerings >, < Benchmark\ or\ app\ code\ under\ test >, < Performance\ metrics >, < Performance\ map\ generated > \} \quad (5)$$

$$F(IaaS) = \{ < EC2, S3, \ldots >, < Benchmarks >, <S_u, E_l, \Phi, B, A, C_o >, < Perf.\ Map > \} \quad (6)$$

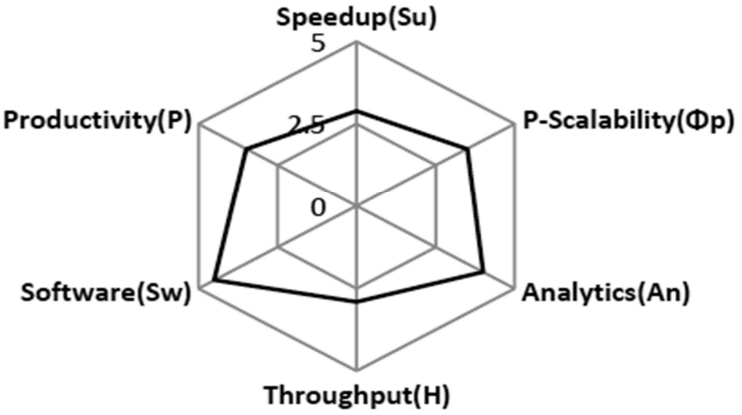
PaaS Cloud Performance

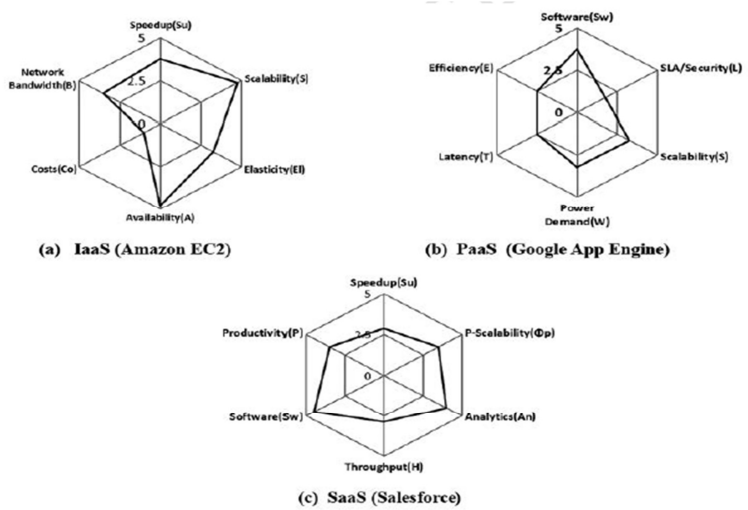
$$F\ (Platform\ Cloud) = \{ < PaaS >, < Apps\ Development, TaaS >, < E, S >, < B, S_w >, < W, L > \}$$



SaaS Cloud Pereformance Model

$$F\,(Application\ Cloud) = \{ < SaaS >, < Marketing,\ Social\ Media >, < Su,\ \Phi_p >, < H,\ S_w, A_n >, < P > \}$$





(a) IaaS (Amazon EC2) (b) PaaS (Google App Engine)

(c) SaaS (Salesforce)

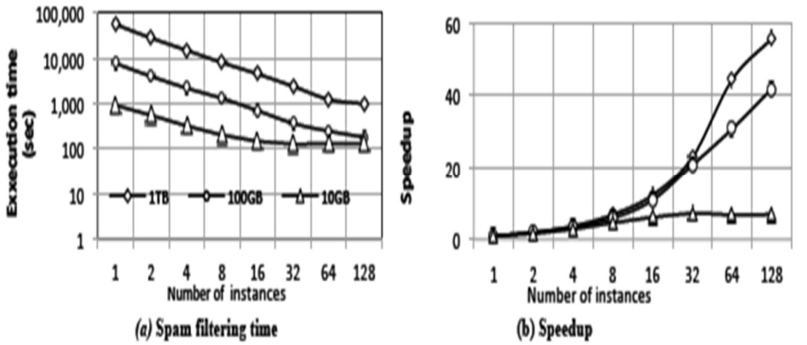
Figure 10.4
Performance maps of various clouds, where data points are extracted from reported Amazon EC2, Google App Engine, and Salesforce clouds. (Courtesy of Hwang et al.,"Cloud Performance Modeling with Benchmark Evaluation of Elastic Scaling Strategies," *IEEE Transactions on Parallel and Distributed Systems*, January 2016.)

Cloud Benchmarks

Table 10.1
Cloud benchmarks, workloads, metrics applied, and systems tested

Benchmark and Developer	Reported Applications and Workloads	Performance Metrics	Clouds Applied and Workload Generation
BenchCloud at USC	Social media applications with big data processing	Speedup, efficiency, QoS, scalability	AWS EC2, Twitter API-workload
CloudSuite at EPFL, Lausanne	Data/graphics analytics, media streaming, and web services	Latency, WIPS, speedup, efficiency, scalability	AWS, GAE, Faban workload generator
HiBench at Intel	Terasort, word count, DFSIO, Nutch indexing, page rank, etc.	Speed, HDFS bandwidth, utilizations (CPU, memory, IO)	Hadoop Random Text Writer, k-Means Data Set
TPC-W by Trans. Proc. Council	Web search and analytical query processing	WIPS, $/WIPS, TPS (transactions per second), QoS, efficiency	AWS EC2, Rackspace, TPC client workload
YCSB by Yahoo!	Synthetic workload, data services	Latency, throughput, speedup, scalability, replication impact	Microsoft Azure, AWS, HBase, Shared MySQL

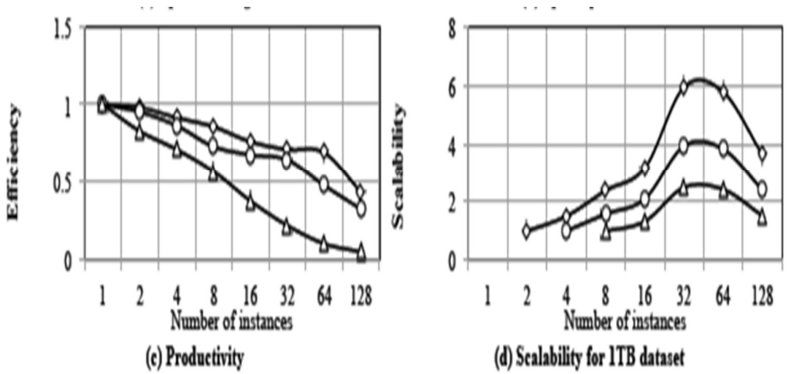
Twitter Spam Filtering Results on EC2



(a) Spam filtering time (b) Speedup

Figure 5. Scale-out BenchClouds results on MapReduce filtering twitter spams over AWS EC2 of various sizes. Parts (a, b, c) apply the same legend. Part &d) shows both scalability measures by scaling from 3 distinct instances.

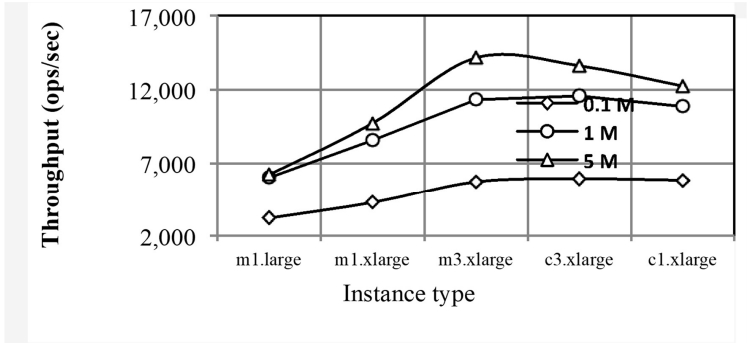
Twitter Spam Filtering on EC2



(c) Productivity (d) Scalability for 1TB dataset

Figure 5. Scale-out BenchClouds results on MapReduce filtering twitter spams over AWS EC2 of various sizes. Parts (a, b, c) apply the same legend. Part &d) shows both scalability measures by scaling from 3 distinct instances.

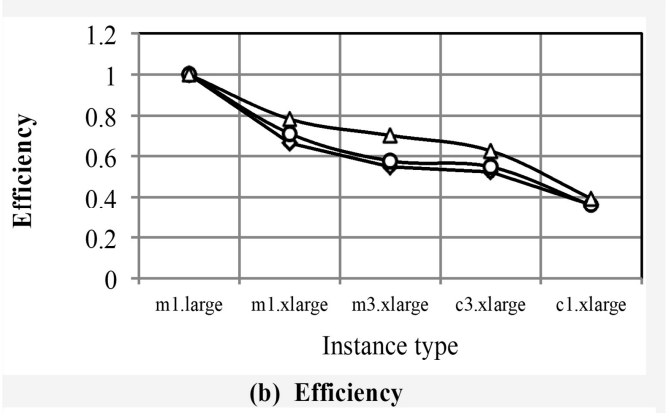
HiBench Results on EC2 over Scale-Up Workloads



(a) Throughput

Figure 8. Scale-up performance of Yahoo! YCSB on EC2 over increasing workload from 100K to 5 M memory-access operations, where the same legend in Part (a) applies in all Parts. All instance types are specified in Table 3.

HiBench Results on EC2 over Scale-Up Workloads



(b) Efficiency

Figure 8. Scale-up performance of Yahoo! YCSB on EC2 over increasing workload from 100K to 5 M memory-access operations, where the same legend in Part (a) applies in all Parts. All instance types are specified in Table 3.

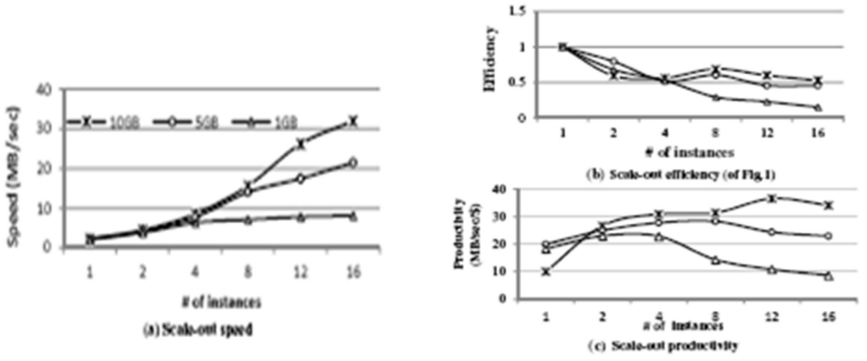


Figure 1. Scale-out performance of HiBench on EC2 virtual clusters built with 1 to 16 *m1.small* machine instances. The three curves correspond to 3 workload sizes executed by WordCount.

Performance of HiBench Scale-Out Experiments

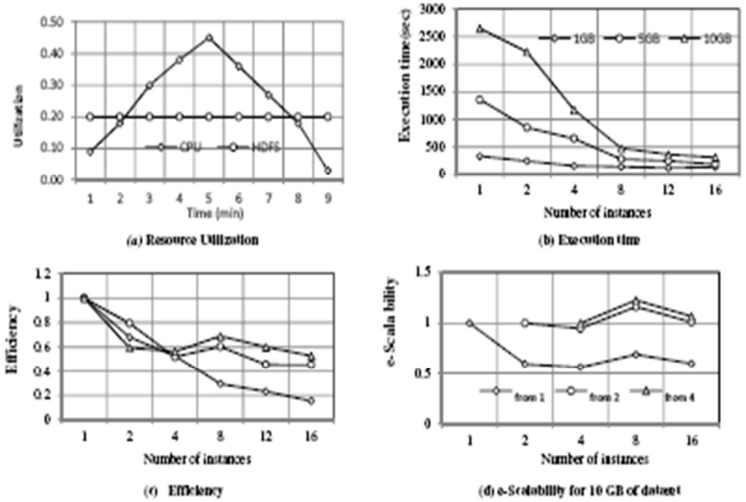
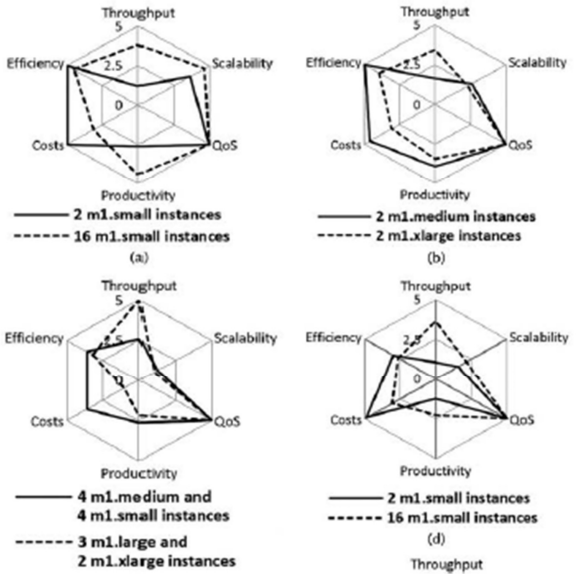


Figure 6. Scale-out performance of HiBench-sort benchmark on Amazon EC2 cloud increasing workload. Parts (b, c) apply the same legend as shown in Part (b). Part(d) scales out from 1, 2 or 4 instances

Scale-Out vs. Scale-Up Performance



Cloud Performance for Mixed Workload

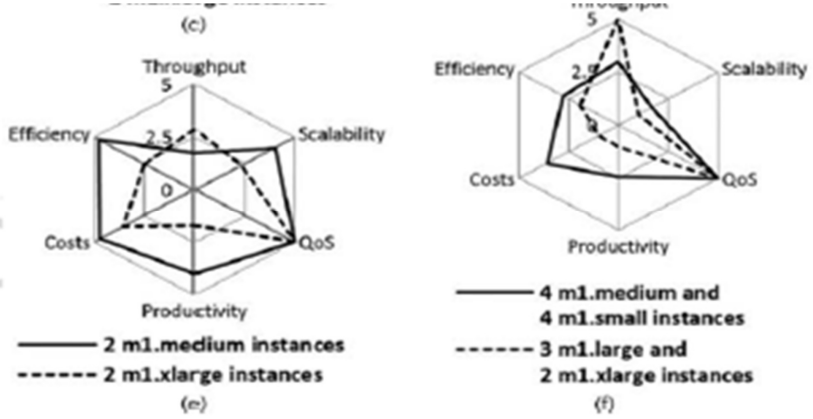


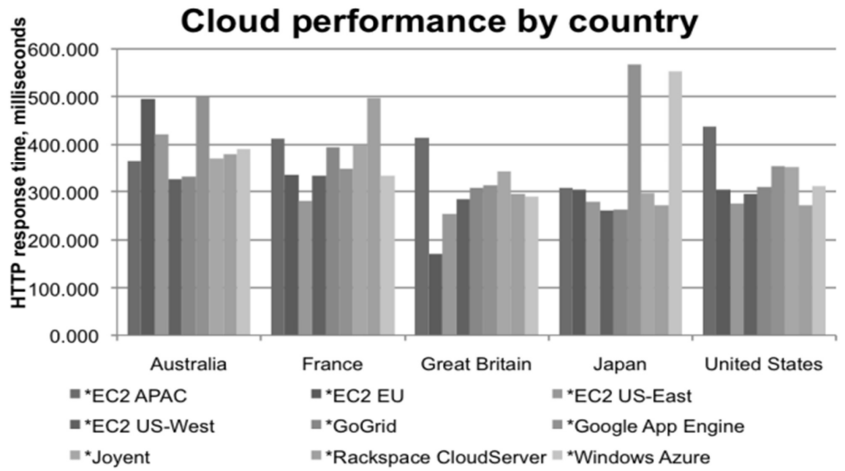
Table 10.4
Scaling techniques based on HiBench benchmarking findings on the EC2

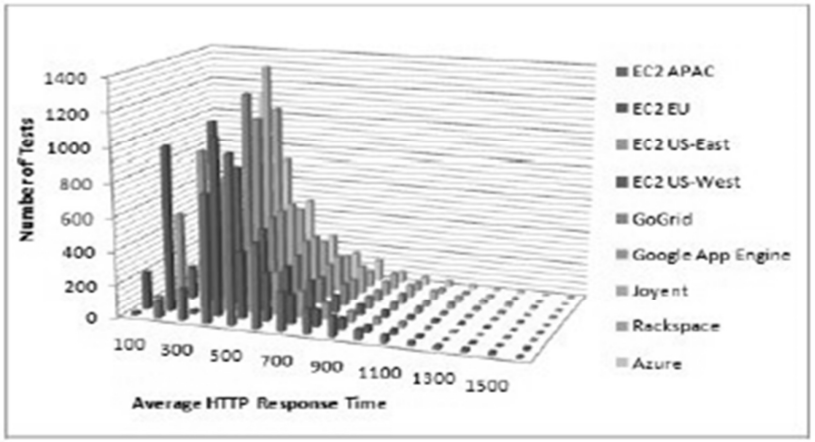
Impact Factors	Scale-Out Technique	Scale-Up Technique	Mixed Scaling
Elasticity Speed, Scaling Complexity, and Overhead	Fast elasticity, possibly supported by auto-scaling and heuristics	High overhead to reconfigure and cannot support auto-scaling	Most difficult to scale with wide range of machine instances
Effects on Performance, Efficiency, and Scalability	Expect scalable performance if the application can exploit parallelism	Switching among heterogeneous nodes may reduce scalability	Flexible app, low efficiency and resource utilization
QoS, Costs, Fault Recovery, and Cloud Productivity	Cost the least, easy to recover, incremental productivity	More cost-effective, but reduced QoS may weaken productivity	High costs, difficult to recover, expect the highest productivity

Table 2 Over-all Performance Demonstrated by Polygon Area on Radar Charts in Fig.5

Scale-Out Mode (Figs.5a, d)	*Cluster Configurations*	*2 small nodes*	*16 small nodes*
	WordCount	34.53	46.85
	Sort	17.02	23.65
Scale-Up Mode (Figs.5b, e)	*Cluster Configurations*	*2 medium nodes*	*2 xlarge nodes*
	WordCount	37.25	31.42
	Sort	41.84	21.22
Mixed Scaling Mode (Figs. 5c, f)	*Cluster Configurations*	*4 medium and 4 small*	*3large and 2 xlarge*
	WordCount	23.39	18.28
	Sort	22.81	11.90

Figure 3 An example radar chart for expressing 11 concerns by 6 cloud user groups (Courtest by Bittorrent, Inc. 2010 [12])





Average response times in ms of 9 cloud service reported in Bitcurrent Report (Courtesy of A. Croll [27])

HTTP Response Time of Working Clouds

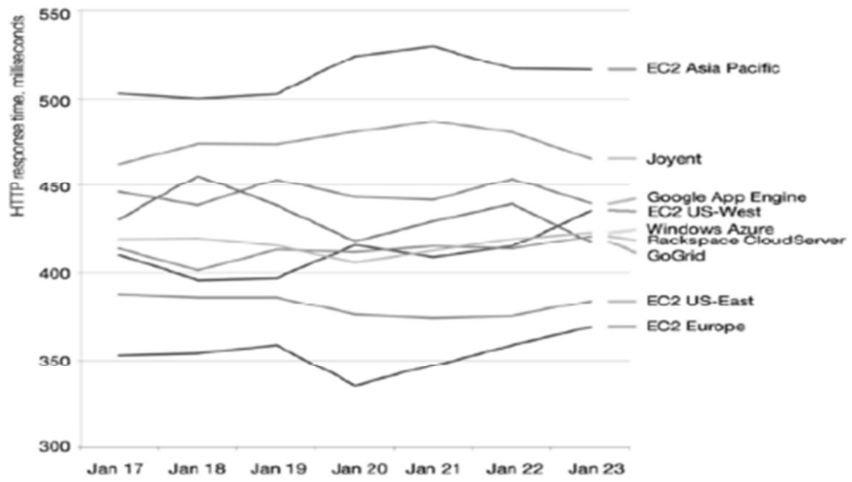
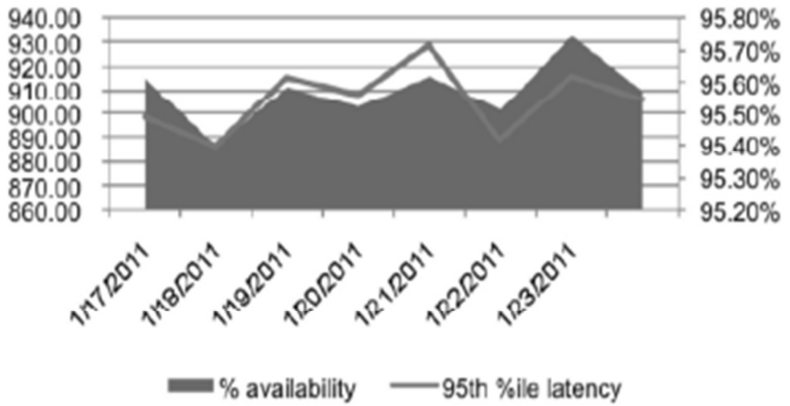
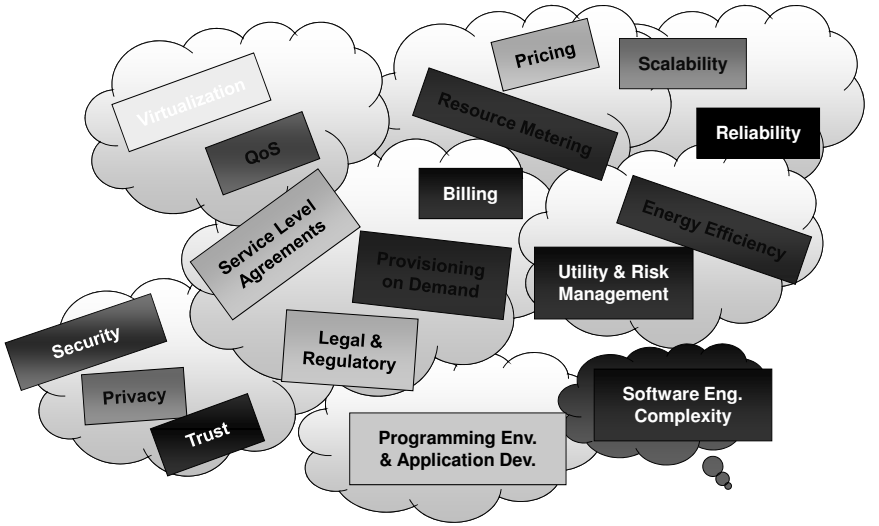


Figure 7 Average HTTP response times in ms of nine cloud service sites in various regions of the world (Courtesy of Bitcurrent 2011, [19])



(a) Average performance and availability of 9 cloud providers

Technical Challenges in Cloud Computing



Some Observations :

- From Figure 1(a), we see a close correlation between cloud latency and availability. Highly available clouds tend to have longer access latency diue to added software overhead.
- The clouds in US and UK are faster. The clouds in Australia and France are slower. GoGrid and Azure have high latenct in Japan. One possible reason is due to not only cloud speed, but trhe network bandwidth, etc.
- Cloud applications in scientific areas cannot win over the Top 500 supercomputers for using commodity hardware, software and network components (not custom-designed).