Autonomic Management of Cost, Performance, and Resource Uncertainty for Deployment of Applications to Infrastructure-as-a-Service (IaaS) Clouds

Wes J. Lloyd
October 22, 2016

Institute of Technology, University of Washington- Tacoma

Outline

- **Introduction**
  - Challenges
  - Background
  - Research Questions
- Methodology
- Research Results
  - Performance Modeling for Component Composition
  - Noisy Neighbor Detection
  - Workload Cost Prediction Methodology
- Summary
- Future Directions

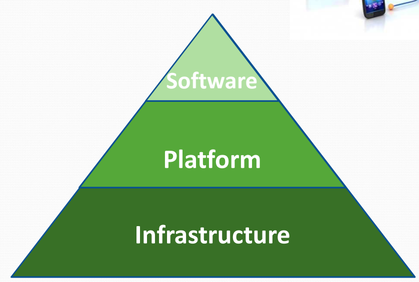
2

Cloud Computing NIST General Definition

"Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (networks, servers, storage, applications and services) that can be rapidly provisioned and reused with minimal management effort or service provider interaction"…

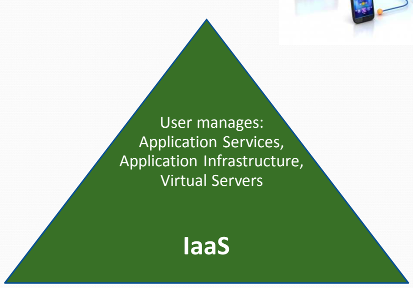
3

Cloud Computing Stack



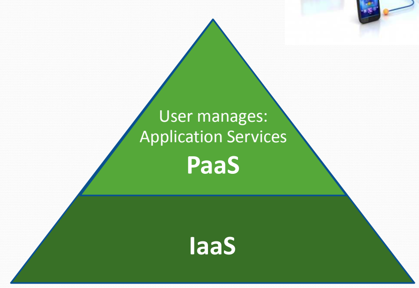
4

Cloud Computing Stack



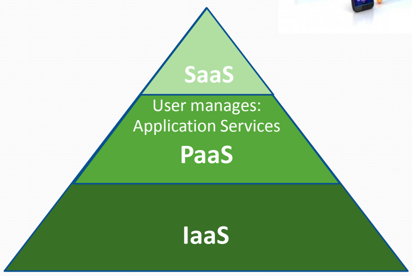
5

Cloud Computing Stack



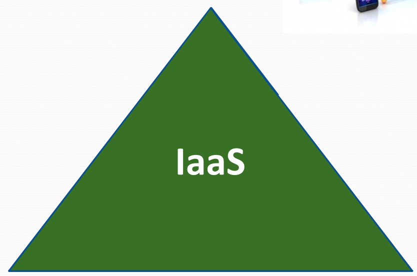
6

Cloud Computing Stack



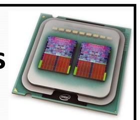
7

Cloud Computing Stack

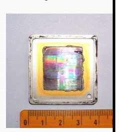


8

Microprocessors Advancements

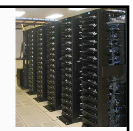


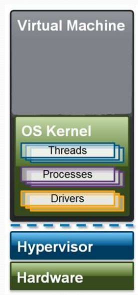
- Smaller die sizes (microns)
  - Lower voltages
  - Improved heat dissipation
  - Energy conservation
  - More transistors, but with similar clock rates
- How do we harness this new transistor density?
  - Multicore CPUs
  - Improve computational throughput
- How do we utilize many-core processors?



9

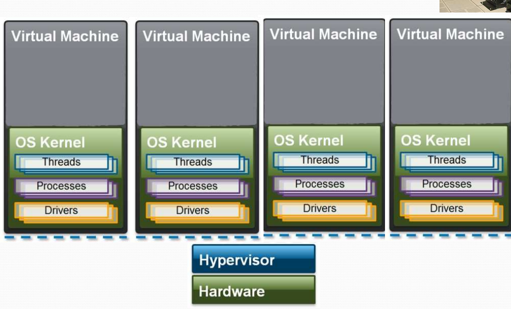
Virtualization





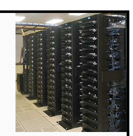
10

Virtualization



11

Containerization



Virtualization

Containerization

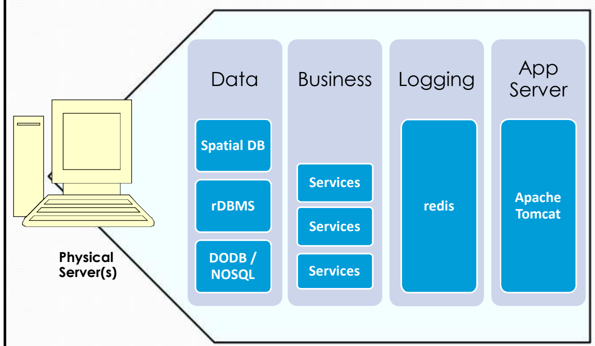
12

Public Cloud Example: Netflix

- Amazon Elastic Compute Cloud (EC2)
  - Continuously run 20,000 to 90,000 VM instances
  - Across 3 regions
  - Host 100s of microservices
  - Process over 100,000 requests/second
  - Host over 1 billion hours of monthly content

13

Traditional Application Deployment



Outline

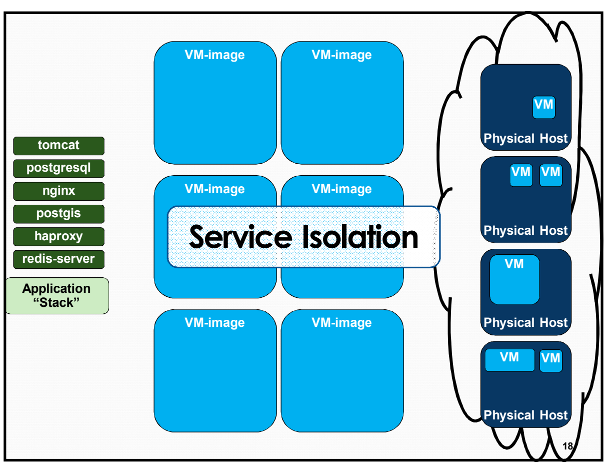
- **Introduction**
  - **Challenges**
  - Background
  - Research Challenges
- Methodology
- Research Results
  - Performance Modeling for Component Composition
  - Noisy Neighbor Detection
  - Workload Cost Prediction Methodology
- Summary
- Future Directions

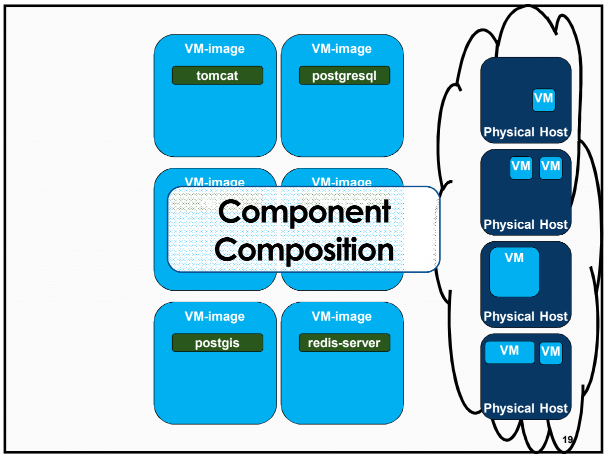
16

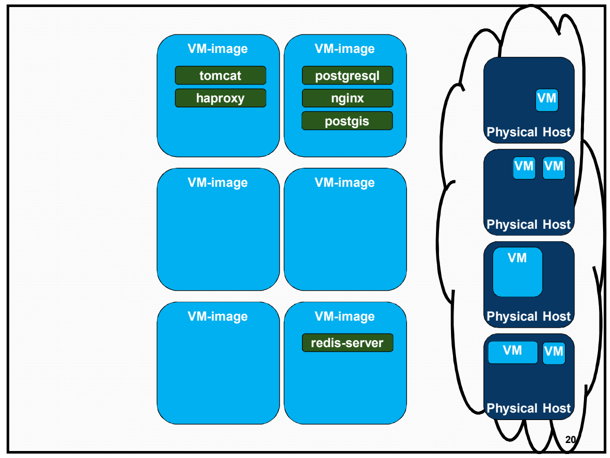
Research Challenges – WHERE

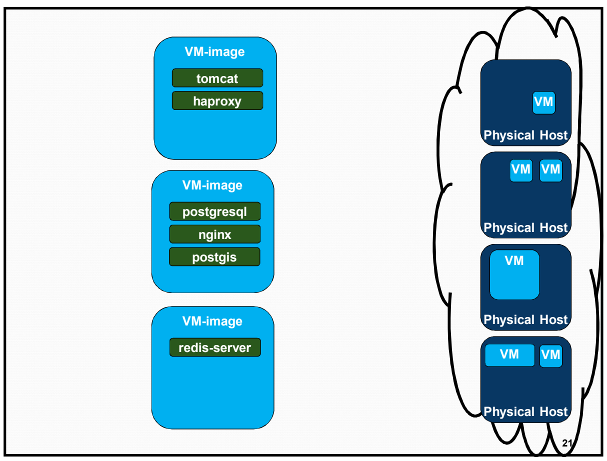
Where should we provision?

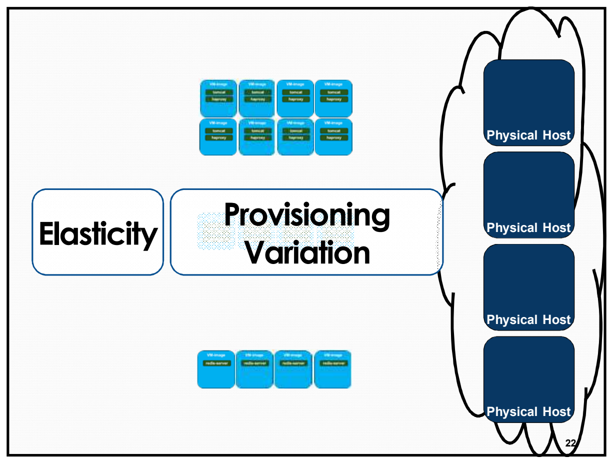
17

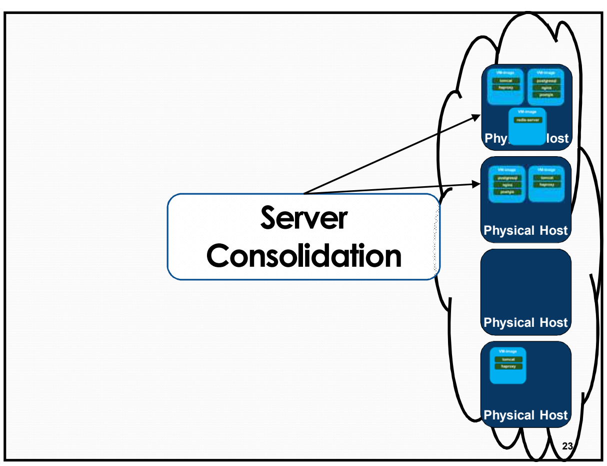


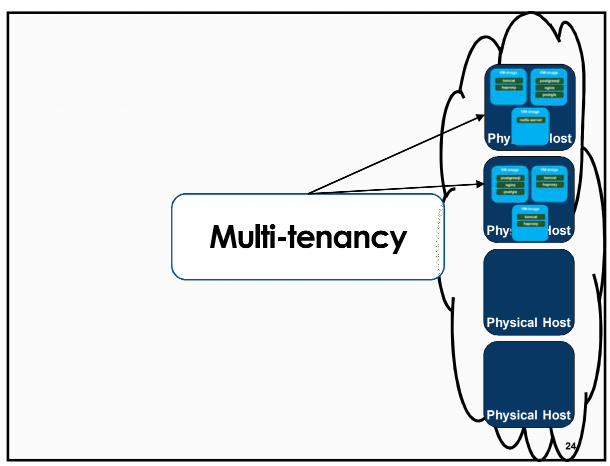

VM-image
tomcat
VM-image
postgresql
Component Composition
VM-image
postgis
VM-image
redis-server
VM
Physical Host
VM VM
Physical Host
VM
Physical Host
VM VM
Physical Host
19


VM-image
tomcat
haproxy
VM-image
postgresql
nginx
postgis
VM-image
VM-image
VM-image
VM-image
redis-server
VM
Physical Host
VM VM
Physical Host
VM
Physical Host
VM VM
Physical Host
20

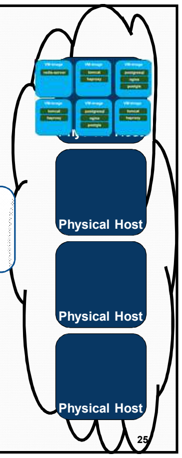

VM-image
tomcat
haproxy
VM-image
postgresql
nginx
postgis
VM-image
redis-server
VM
Physical Host
VM VM
Physical Host
VM
Physical Host
VM VM
Physical Host
21


Elasticity
Provisioning Variation
Physical Host
Physical Host
Physical Host
Physical Host
22


Server Consolidation
Physical Host
Physical Host
Physical Host
23


Multi-tenancy
Physical Host
Physical Host
24

Overprovisioning



25

Research Challenges – WHERE

Service Isolation

Component Composition

Provisioning Variation

Server Consolidation

Multi-tenancy

Overprovisioning

Resource Contention

26

Research Challenges - WHAT

What should we provision?

Performance

53+ types

27

Research Challenges - WHAT

Size
Vertical Scaling

Quantity
Horizontal Scaling

28

Research Challenges - WHAT

Size
Vertical Scaling

Quantity
Horizontal Scaling

Amazon
VM types
m1.large
c3.xlarge
m2.2xlarge
m4.2xlarge
c1.xlarge
m3.medium
m2.4xlarge
d2.8xlarge
m1.xlarge
c3.large

53+ types

Qualitative
Resource descriptions

Virtualization
Hypervisors

29

Research Challenges - WHAT

Size
Vertical Scaling

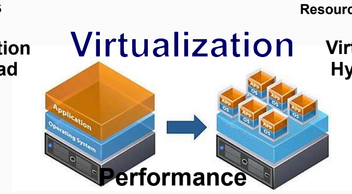
Quantity
Horizontal Scaling

Amazon
VM types

Qualitative
Resource descriptions

Virtualization
Overhead

Virtualization
Hypervisors



30

Research Challenges - WHAT

Size
Vertical Scaling

Quantity
Horizontal Scaling

Amazon
VM types

Qualitative
Resource descriptions

Virtualization
Overhead

Virtualization
Hypervisors

Performance

31

Research Challenges - WHEN

When should we provision?

32

Research Challenges - WHEN

Hot Spot Detection | **Provisioning Latency**

Future Load Prediction | **Pre-provisioning**

WHEN

33

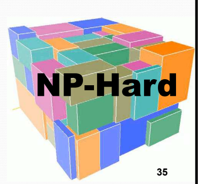
Outline

- **Introduction**
  - Challenges
  - **Background**
  - Research Questions
- Methodology
- Research Results
  - Performance Modeling for Component Composition
  - Noisy Neighbor Detection
  - Workload Cost Prediction Methodology
- Summary
- Future Directions

34

Virtual Machine (VM) Placement as "Bin Packing Problem"

- Components *items* → virtual machines (VMs) *bins*
- Virtual machines (VMs) *items* → physical machines (PMs) *bins*
- Dimensions
  - # CPU cores, CPU clock speed, architecture
  - RAM, hard disk size, # cores
  - Disk read/write throughput
  - Network read/write throughput
- PM capacities vary dynamically
- VM resource utilization varies
- Component requirements vary



35

Why Gaps Exist

- Public clouds
  - Research is time/cost prohibitive
  - Hardware abstraction: Users are not in control
  - Rapidly changing system implementations
- Private clouds:
  - Wide variance of implementations
  - Systems continuously evolve
- Performance modeling (large problem space)
- Virtualization misunderstood or overlooked

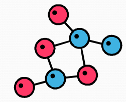
36

Outline

- **Introduction**
  - Challenges
  - Background
  - **Research Questions**
- Methodology
- Research Results
  - Performance Modeling for Component Composition
  - Noisy Neighbor Detection
  - Workload Cost Prediction Methodology
- Summary
- Future Directions

37

Research Questions (1/2)



RQ-1: Component composition

How does resource utilization and *service oriented application* (SOA) performance vary relative to component composition across VMs?

RQ-2: Performance modeling

Which resource utilization variables and modeling techniques best help predict SOA performance?

38

Research Questions (2/2)

RQ-3: Noisy neighbors

What performance implications result from resource contention and how can we avoid it?

RQ-4: Infrastructure prediction

How can we predict the required cloud infrastructure to satisfy performance requirements for SOA workload hosting?

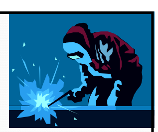
39

Outline

- Introduction
  - Challenges
  - Background
  - Research Questions
- **Methodology**
- Research Results
  - Performance Modeling for Component Composition
  - Noisy Neighbor Detection
  - Workload Cost Prediction Methodology
- Summary
- Future Directions

40

Methodology



- Benchmark Workloads
  - Scientific Modeling Workloads
- Profile resource utilization
  - Collect VM-level data
- Analytics: construct performance and cost models
  - R: statistical regression, neural networks
- Evaluate and refine models
  - Develop heuristics

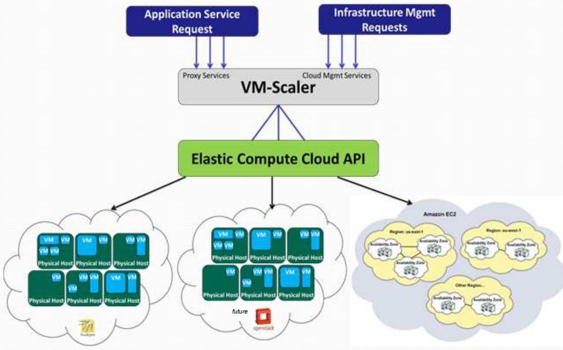
41

Scientific Modeling Workloads

- USDA Cloud Services Integration Platform (CSIP):
  - Framework for scientific modeling-as-a-service
- Scientific modeling SOAs:
  - RUSLE2 – Soil erosion model
  - WEPS – Wind Erosion Prediction System
  - SWAT-DEG: Stream channel degradation prediction
    Monte carlo workloads
  - Comprehensive Flow Analysis tools
    Load estimator, Load duration curve, Flow duration Curve, Baseflow, Flood analysis, Drought analysis

42

VM-Scaler

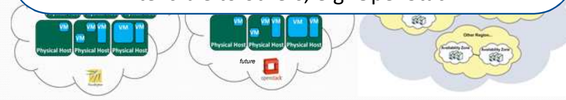


43

VM-Scaler

Application Service Request | Infrastructure Mgmt Requests

- REST/JSON Web services application
  - Harnesses Amazon's EC2 API
  - Provides cloud infrastructure management
  - Supports scientific modeling-as-a-service
  - Supports research and IaaS experimentation
  - Supports Amazon, Eucalyptus 3/4 clouds
  - Extensible to others, e.g. OpenStack



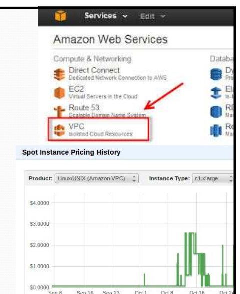
44

Eucalyptus Private Clouds

- Implemented (3) Private Clouds @ Colo State
- Eramscloud: 10 x Oracle X6270 blade system
  - Dual Intel Xeon 4core HT 2.8 GHz CPUs
  - 72 GB ram, 4 x 600 GB 15k rpm HDDs
  - CentOS 5/6 x86_64 (host OS)
  - Ubuntu x86_64 (guest OS)
- Eucalyptus 3/4
  - Amazon EC2 API support
  - Nodes(NC), Cloud(CLC), Cluster(CC), Storage(SC)
  - Managed mode networking with private VLANs
  - XEN/KVM hypervisors, para/full virtualization

45

Amazon AWS



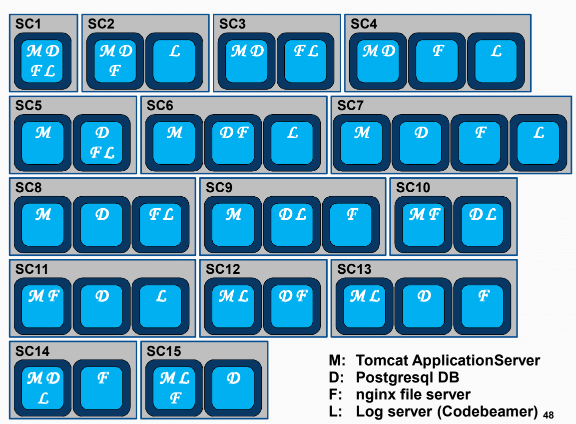
- Spot instances
- Virtual Private Cloud (VPC)
- Ubuntu (guests)
  - Xen virtualization
- Many VM types and generations
  - m1.medium, m1.large, m1.xlarge, c1.medium, c1.xlarge
  - m2.xlarge, m2.2xlarge, and m2.4xlarge
  - c3.large, c3.xlarge c3.2xlarge, m3.large

46

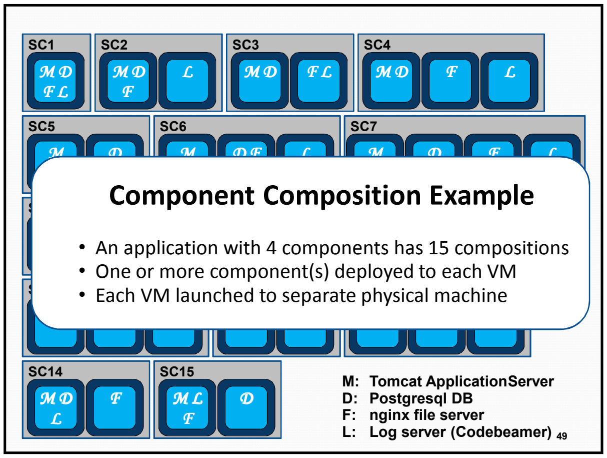
Outline

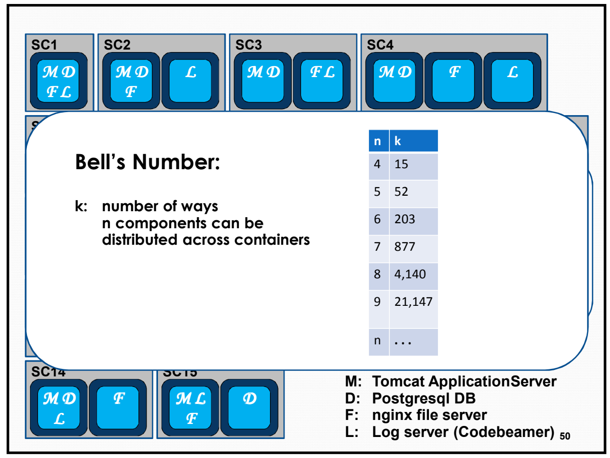
- Introduction
  - Challenges
  - Background
  - Research Questions
- Methodology
- **Research Results**
  - **Performance Modeling for Component Composition**
  - Noisy Neighbor Detection
  - Workload Cost Prediction Methodology
- Summary
- Future Directions

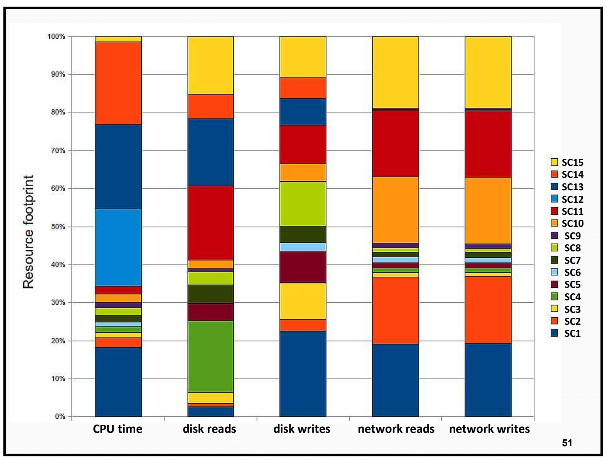
47

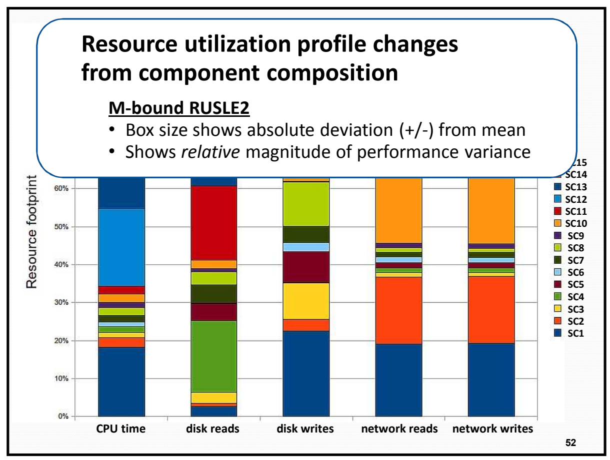


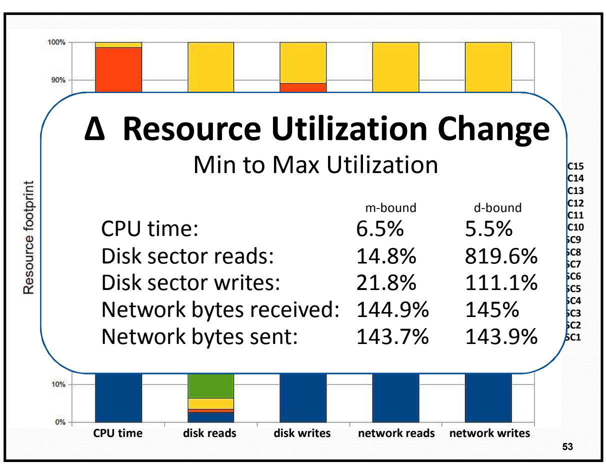
48

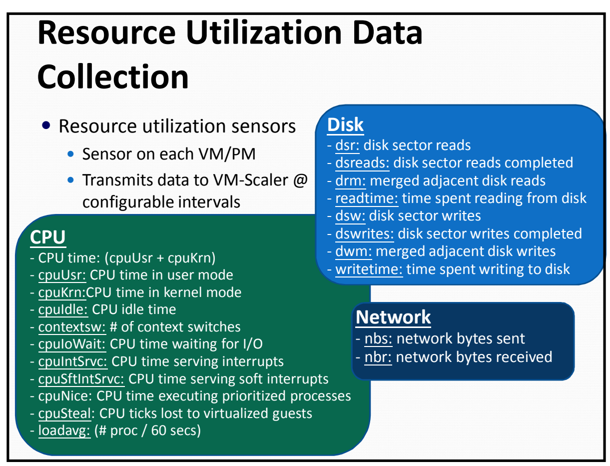

SC1
SC2
SC3
SC4
SC5
SC6
SC7
SC14
SC15
Component Composition Example
• An application with 4 components has 15 compositions
• One or more component(s) deployed to each VM
• Each VM launched to separate physical machine
M: Tomcat ApplicationServer
D: Postgresql DB
F: nginx file server
L: Log server (Codebeamer)
49


SC1
SC2
SC3
SC4
SC14
Bell's Number:
k: number of ways
n components can be
distributed across containers
n k
4 15
5 52
6 203
7 877
8 4,140
9 21,147
n . . .
M: Tomcat ApplicationServer
D: Postgresql DB
F: nginx file server
L: Log server (Codebeamer)
50

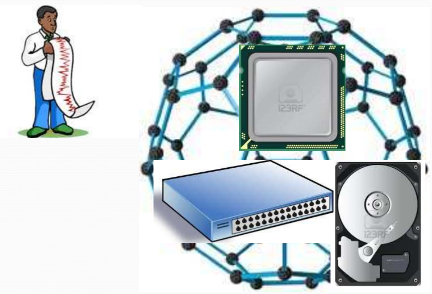

Resource footprint
CPU time
disk reads
disk writes
network reads
network writes
SC15
SC14
SC13
SC12
SC11
SC10
SC9
SC8
SC7
SC6
SC5
SC4
SC3
SC2
SC1
51


Resource utilization profile changes
from component composition
M-bound RUSLE2
• Box size shows absolute deviation (+/-) from mean
• Shows *relative* magnitude of performance variance
Resource footprint
CPU time
disk reads
disk writes
network reads
network writes
52


Δ Resource Utilization Change
Min to Max Utilization
m-bound
d-bound
CPU time: 6.5% 5.5%
Disk sector reads: 14.8% 819.6%
Disk sector writes: 21.8% 111.1%
Network bytes received: 144.9% 145%
Network bytes sent: 143.7% 143.9%
Resource footprint
CPU time
disk reads
disk writes
network reads
network writes
53


Resource Utilization Data Collection
• Resource utilization sensors
• Sensor on each VM/PM
• Transmits data to VM-Scaler @ configurable intervals
Disk
- dsr: disk sector reads
- dsreads: disk sector reads completed
- drm: merged adjacent disk reads
- readtime: time spent reading from disk
- dsw: disk sector writes
- dswrites: disk sector writes completed
- dwm: merged adjacent disk writes
- writetime: time spent writing to disk
CPU
- CPU time: (cpuUsr + cpuKrn)
- cpuUsr: CPU time in user mode
- cpuKrn:CPU time in kernel mode
- cpuIdle: CPU idle time
- contextsw: # of context switches
- cpuIoWait: CPU time waiting for I/O
- cpuIntSrvc: CPU time serving interrupts
- cpuSftIntSrvc: CPU time serving soft interrupts
- cpuNice: CPU time executing prioritized processes
- cpuSteal: CPU ticks lost to virtualized guests
- loadavg: (# proc / 60 secs)
Network
- nbs: network bytes sent
- nbr: network bytes received

Can Resource Utilization Statistics Model Application Performance?

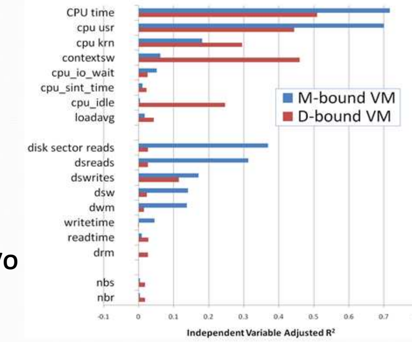


Which resource utilization variables are the best predictors?

CPU

Disk I/O

Network I/O



Which modeling techniques were most effective?

- Multiple Linear Regression (MLR)
- Stepwise Multiple Linear Regression (MLR-step)
- Multivariate Adaptive Regression Splines (MARS)
- Artificial Neural Network (ANNs)

Which modeling techniques were most effective?

	Model	Type	Adj. R^2	(ms) $RMSE_{train}$	(ms) $RMSE_{test}$
Multiple Linear Regression	D-bound	MLR	0.9107	4532.85	44904
	M-bound	MLR	0.8546	616.98	807.34
Stepwise MLR	D-bound	MLR-step	0.9118	4589.27	43919
	M-bound	MLR-step	0.8571	621.41	799.22
Multivariate Adaptive Regression Splines	D-bound	MARS	0.918	4472.32	45137
	M-bound	MARS	0.8718	596.45	825.34
Artificial Neural Network	D-bound	ANN	n/a	4440.03	44094
	M-bound	ANN	n/a	595.49	800.71

Which modeling techniques were most effective?

Model performance did not vary much

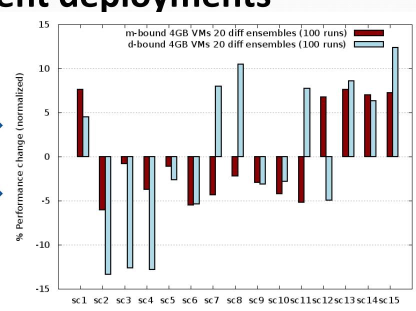
Best vs. Worst

D-Bound		M-Bound
.11%	$RMSE_{train}$	.08%
.89%	$RMSE_{test}$	.08%
.40	rank err	.66

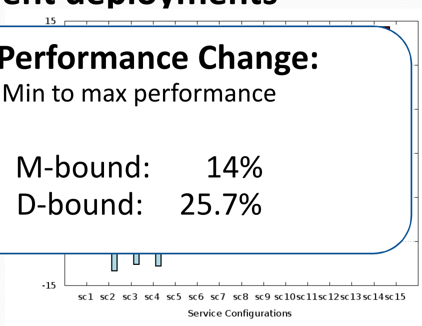
Performance implications of component deployments

Slower deployments

Faster deployments



Performance implications of component deployments



Δ Performance Change:

Min to max performance

M-bound: 14%

D-bound: 25.7%

61

Outline

- Introduction
  - Challenges
  - Background
  - Research Questions
- Methodology
- **Research Results**
  - Performance Modeling for Component Composition
  - **Noisy Neighbor Detection**
  - Workload Cost Prediction Methodology
- Summary
- Future Directions

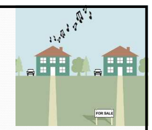
62

CpuSteal

- CpuSteal: VM's CPU core is ready to execute but the physical CPU core is busy
- Symptom of over provisioning physical servers
- Factors which cause CpuSteal:
  1. Processors shared by too many busy VMs
  2. Hypervisor kernel (Xen dom0) is occupying the CPU
  3. VM's CPU time share <100% for 1 or more cores, and 100% is needed for a CPU intensive workload.

63

Noisy Neighbor (NN-Detect) Detection Methodology



- Noisy neighbors cause resource contention and degrade performance of worker VMs
  - Identify noisy neighbors by analyzing *cpuSteal*
- Detection method:

Step 1: Execute processor intensive workload across pool of VMs.

Step 2: Capture total *cpuSteal* for each VM for the workload.

Step 3: Calculate average *cpuSteal* for the workload ($cpuSteal_{avg}$).

Identify NNs using application agnostic and specific thresholds…

64

Amazon EC2 *CpuSteal* Analysis

VM Type	Host CPU Intel Xeon	Average R^2 linear reg.	Average *cpuSteal* per core	% with Noisy Neighbors
us-east-1c				
c3.large-2c	E5-2680v2/10c	.1753	2.35	0%
m3.large-2c	E5-2670v2/10c	-	1.58	0%
m1.large-2c	E5-2650v0/8c	.5568	7.62	12%
m2.xlarge-2c	X5550/4c	.4490	310.25	18%
m1.xlarge-4c	E5-2651v2/12c	.9431	7.25	4%
m3.medium-1c	E5-2670v2/10c	.0646	17683.2[1]	n/a
c1.xlarge-8c	E5-2651v2/12c	.3658	1.86	0%
us-east-1d				
m1.medium-1c	E5-2650v0/8c	.4545	6.2	10%
m2.xlarge-2c	E5-2665v0/8c	.0911	3.14	0%

65

Amazon EC2 *CpuSteal* Analysis

VM Type	Host CPU Intel Xeon	Average R^2 linear reg.	Average *cpuSteal* per core	% with Noisy Neighbors
m1.xlarge-4c	E5-2651v2/12c	.9431	7.25	4%
m3.medium-1c	E5-2670v2/10c	.0646	17683.2[1]	n/a
c1.xlarge-8c	E5-2651v2/12c	.3658	1.86	0%
us-east-1d				
m1.medium-1c	E5-2650v0/8c	.4545	6.2	10%
m2.xlarge-2c	E5-2665v0/8c	.0911	3.14	0%

Test Configuration:

- Completed 4 x 1000 WEPS runs over ~5 hours
- ~50 VM pools (c1.xlarge 25, m3/m1.medium 60)
- Round robin load balancing of runs across pools

66

Amazon EC2 *CpuSteal* Analysis

Key Result #1

4 VM types had $R^2 > 0.44$

m1.large, m2.xlarge, m1.xlarge, m1.medium

Key Result #2

Where *cpuSteal* could not be predicted it did not exist. This hardware tended to be CPU core dense. (e.g. 8, 10, or 12)

67

Noisy Neighbor Performance Degradation

- Compared performance of small 5 VM pools
  - 5 Noisy-Neighbor VMs
  - 5 regular VMs
- WEPS: 10 x 100 runs
- RUSLE2: 10 x 660 runs
- Normalized results to regular VM pools

68

EC2 Noisy Neighbor Performance Degradation

VM type	Region	WEPS	RUSLE2
m1.large E5-2650v0/8c	us-east-1c	117.68% df=9.866 $p=6.847 \cdot 10^{-8}$	125.42% df=9.003 p=.016
m2.xlarge X5550/4c	us-east-1c	107.3% df=19.159 p=.05232	102.76% df=25.34 $p=1.73 \cdot 10^{-11}$
c1.xlarge E5-2651v2/12c	us-east-1c	100.73% df=9.54 p=.1456	102.91% n.s.
m1.medium E5-2650v0/8c	us-east-1d	111.6% df=13.459 $p=6.25 \cdot 10^{-8}$	104.32% df=9.196 $p=1.173 \cdot 10^{-5}$

69

EC2 Noisy Neighbor Performance Degradation

Key Result #1

Maximum performance loss:

WEPS 18%, RUSLE2 25%

Key Result #2

3 VM types with significant performance loss ($p < .05$)

Average performance loss: WEPS/RUSLE2 ~ 9%

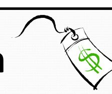
70

Outline

- Introduction
  - Challenges
  - Background
  - Research Questions
- Methodology
- **Research Results**
  - Performance Modeling for Component Composition
  - Noisy Neighbor Detection
  - **Workload Cost Prediction Methodology**
- Summary
- Future Directions

71

Workload Cost Prediction



Example:
Base VM-type: [`5 x c3.xlarge`] = 20 cores

- Scale the number of worker VMs
- Achieve ***equivalent*** performance using any VM type
- Load balance workload across VM pool

`c3.xlarge → c1.medium` `c3.xlarge → m2.2xlarge`

`c3.xlarge → m1.large` `c3.xlarge → m2.xlarge`

`c3.xlarge → m2.4xlarge` `c3.xlarge → m1.xlarge`

`c3.xlarge → m1.medium`

72

Workload Cost Prediction

- Predict number of VMs of alternate type(s) supporting ***equivalent*** workload execution time
  - Execution within +/- 2 seconds using any base VM type
- Supports use of alternate VM types based on
  - Public cloud: lowest price VM-type
  - Private cloud: Most available or convenient VM-type
- Some VM types may be too slow to be viable

73

Approach

- Harness Linux CPU time accounting principles

Workload wall clock time can be calculated:

Sum CPU resource utilization variables across the worker VM pool, and divide by total # of CPU cores

$$\text{Workload}_{time} = \frac{\sum\begin{Bmatrix} cpuUsr_T + cpuKrnT + cpuIdleT + cpuIoWaitT + \\ cpuIntSrvc_T + cpuSoftIntSrvcT + cpuNiceT + cpuStealT \end{Bmatrix}}{VM_{cores}}$$

74

VM-type Resource Variable Conversion Multiple Linear Regression

RU variable	adjusted R^2 m1.xlarge LR		adjusted R^2 m1.xlarge MLR		adjusted R^2 c1.medium MLR
cpuUsr	.9924		.9993		.9983
cpuKrn	.9464	Single Linear Regres.	.989	Multp Linear Regres.	.9784
cpuIdle	.7103		.9674		.9498
cpuIoWait	.9205		.9584		.9725
	adjusted R^2 m2.xlarge MLR				adjusted R^2 m3.xlarge MLR
cpuUsr	.9987				.9992
cpuKrn	.967	Strong predictability forms the crux of the approach			.9831
cpuIdle	.9235				.9554
cpuIoWait	.9472				.9831

75

VM-type Resource Variable Conversion Multiple Linear Regression

RU variable	adjusted R^2 m1.xlarge LR		adjusted R^2 m1.xlarge MLR	adjusted R^2 c1.medium MLR
cpuKrn	.967	Strong predictability forms the crux of the approach		.9831
cpuIdle	.9235			.9554
cpuIoWait	.9472			.9831

High R^2 reflects our ability to use c3.xlarge profiles to predict resource requirements for the same workload on other VM types

76

VM infrastructure predictions for equivalent performance

Mean Absolute Error (# VMs)

SOA / VM-type	PS-1 (RS-2)
WEPS	.5
RUSLE2	.125
SWATDEG-STOC	.5
SWATDEG-DET	.125
m1.xlarge	.25
c1.medium	.5
m2.xlarge	.25
m3.xlarge	.25
Average	***.3125***

77

Workload hosting cost prediction

10,000 compute hours

SOA	m1.xlarge	c1.medium	m2.xlarge
WEPS	$38,400	$22,400	$24,600
RUSLE2	$38,400	$22,400	$24,600
SWATDEG-Stoc	n/a	$19,600	$24,600
SWATDEG-Det	$38,400	$25,200	$28,700
Total	***$115,200***	***$89,600***	***$102,500***

	m3.xlarge	Total error
WEPS	$27,000	-$7,600
RUSLE2	$27,000	$0
SWATDEG-Stoc	$27,000	-$8,600
SWATDEG-Det	$27,000	+$1,300
Total	***$108,000***	***-$14,900 (3.59%)***

78

Workload hosting cost prediction
10,000 compute hours

SOA	m1.xlarge	c1.medium	m2.xlarge

Key Result

Maximum Cost Δ:

~28.6% ($25,600 for 10,000 hours)

m1.xlarge (4-core VM) vs. c1.medium (2-core VM)

SWATDEG-Stoc	$27,000	-$8,600
SWATDEG-Det	$27,000	+$1,300
Total	***$108,000***	***-$14,900 (3.59%)***

Outline

- Introduction
  - Challenges
  - Background
  - Research Questions
- Methodology
- Research Results
  - Performance Modeling for Component Composition
  - Noisy Neighbor Detection
  - Workload Cost Prediction Methodology
- **Summary**
- Future Directions

Retrospective

- Infrastructure-as-a-service leads to the simplistic view that resource are homogeneous and scaling can infinitely provide linear performance gains
- This research has demonstrated many infrastructure management challenges in cloud computing
- Our results provide:

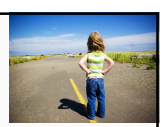
  Methodologies and analytics to support application performance improvements while reducing infrastructure hosting costs

Enabling us to do *more* with less!

Outline

- Introduction
  - Challenges
  - Background
  - Research Questions
- Methodology
- Research Results
  - Performance Modeling for Component Composition
  - Noisy Neighbor Detection
  - Workload Cost Prediction Methodology
- Summary
- **Future Directions**

Future Directions (1/5)



- Optimizing performance and cost using new workloads
  - Bioinformatics (Yeung-Rhee)
  - Machine Learning (DeCock)
  - Geospatial (Ali)
  - Cyber-Physical IoT (Tolentino)
  - Big Data analytic workloads (Teredesai)
  - eScience Institute (UW Seattle)
- Heavy I/O, Heavy processing, Long lifetime
- Infrastructure management improvements for **Big Data** system performance

Future Directions (2/5)

- Characterize different technologies
  - Harness performance modeling
  - Support tool development:
    - ➡ What is the best infrastructure for my workload?
    - ➡ What is the cost of deployment?
  - Docker, CoreOS/Rocket, KVM, XEN
- Cost and performance of IaaS, PaaS, SaaS
  - ➡ What service level is best for my workload?

Future Directions (3/5)

- Large scale public cloud resource contention study
  - What trends and usage patterns emerge over time?
  - How can we harness cloud usage data to best improve application performance while reducing hosting costs?

85

Future Directions (4/5)

- Continuous application deployment
  - Reactive component composition
  - Using OS containers (Docker, LXC)
  - How can deployments adapt to to resource contention?

86

Future Directions (5/5)

- Harness and develop hybrid, federated, mobile, and ad-hoc cloud infrastructures
  - To build resilient, scalable infrastructures using heterogeneous devices (IoT)
  - How do we transparently provide resource elasticity, workload migration, and high availability with diverse clouds to end users?
- Support green computing goals:
  - Opportunistic workload consolidation and migration to the most sustainable, economical, and energy efficient resources

87

Publications: Journal

1. W. Lloyd, S. Pallickara, O. David, M. Arabi, and K. W. Rojas, "Demystifying the Clouds: Harnessing Resource Utilization Models for Cost Effective Infrastructure Alternatives" IEEE Transactions on Cloud Computing Journal, 2016, In Press.
2. O. David, J. C. Ascough II, W. Lloyd, T. R. Green, K. W. Rojas, G. H. Leavesley, and L. R. Ahuja, "A software engineering perspective on environmental modeling framework design: The Object Modeling System," Environmental Modeling & Software, 39 (1): 201–213. 2013. Elsevier. (4.42 Impact Factor 2014)
3. W. Lloyd, S. Pallickara, O. David, J. Lyon, M. Arabi, and K. W. Rojas, "Performance implications of multi-tier application deployments on Infrastructure-as-a-Service clouds: Towards performance modeling," Future Generation Computer Systems, 29 (5): 1254-1264. 2013. Elsevier. (2.786 Impact Factor)
4. W. Lloyd, O. David, J. Ascough, K. Rojas, J. Carlson, G. Leavesley, P. Krause, T. Green, L. Ahuja, Elsevier, Environmental Modeling & Software, 26 (10): 1240-1250. 2011. Elsevier. (4.42 Impact Factor 2014)
5. A. Dozier, O. David, M. Arabi, W. Lloyd, Y. Zhang, A minimally invasive model data passing interface for integrating legacy environmental system models. Submitted to Elsevier Environmental Modeling & Software, Accepted for publication. (4.42 Impact Factor 2014)
6. W. Lloyd, S. Pallickara, O. David, M. Arabi, and K. W. Rojas, "Improving VM Placements to Mitigate Resource Contention and Heterogeneity in Cloud Settings for Scientific Modeling Services", *In preparation*.

88

Publications: Conference

1. W. Lloyd, S. Pallickara, O. David, M. Arabi, and K. W. Rojas, "Dynamic Scaling for Service Oriented Applications: Implications of Virtual Machine Placement on IaaS Clouds," in Proceedings of the 2014 IEEE International Conference on Cloud Engineering (IC2E '14), 2014. (20.9% acceptance rate)
2. W. Lloyd, O. David, M. Arabi, J. C. Ascough II, T. R. Green, J. Carlson, and K. W. Rojas, "The Virtual Machine (VM) Scaler: An Infrastructure Manager Supporting Environmental Modeling on IaaS Clouds," in Proceedings iEMSs 2014 International Congress on Environmental Modeling and Software, p. 8.
3. O. David, W. Lloyd, K. W. Rojas, M. Arabi, F. Geter, J. Carlson, G. H. Leavesley, J. C. Ascough II, and T. R. Green, "Model as a Service (MaaS) using the Cloud Service Innovation Platform (CSIP)," in Proceedings iEMSs 2014 International Congress on Environmental Modeling and Software, p. 8.
4. T. Wible, W. Lloyd, O. David, and M. Arabi, "Cyberinfrastructure for Scalable Access to Stream Flow Analysis," in Proceedings iEMSs 2014 International Congress on Environmental Modeling and Software2, p. 6.
5. W. Lloyd, S. Pallickara, O. David, J. Lyon, M. Arabi, and K. W. Rojas, "Service isolation vs. consolidation: Implications for IaaS cloud application deployment," in Proceedings of the IEEE International Conference on Cloud Engineering, IC2E 2013, 2013, pp. 21–30. (20.5% acceptance rate)
6. W. Lloyd, S. Pallickara, O. David, J. Lyon, M. Arabi, and K. W. Rojas, "Performance modeling to support multi-tier application deployment to infrastructure-as-a-service clouds," in Proceedings - 2012 IEEE/ACM 5th International Conference on Utility and Cloud Computing, UCC 2012, 2012, pp. 73–80. (27% acceptance rate)
7. W. Lloyd, O. David, J. Lyon, K. W. Rojas, J. C. Ascough II, T. R. Green, and J. Carlson, "The Cloud Services Innovation Platform - Enabling Service-Based Environmental Modeling Using IaaS Cloud Computing," in Proceedings iEMSs 2012 International Congress on Environmental Modeling and Software, 2012, p. 8.
8. W. Lloyd, S. Pallickara, O. David, J. Lyon, M. Arabi, and K. W. Rojas, "Migration of multi-tier applications to infrastructure-as-a-service clouds: An investigation using kernel-based virtual machines," Proc. - 2011 12th IEEE/ACM Int. Conf. Grid Comput. Grid 2011, pp. 137–144, 2011. (29% acceptance rate)

89

Questions

90