Autonomic Resource Management for Multi-tier Application Deployment to Infrastructure-as-a-Service (IaaS) Clouds

PHD Dissertation Proposal Defense

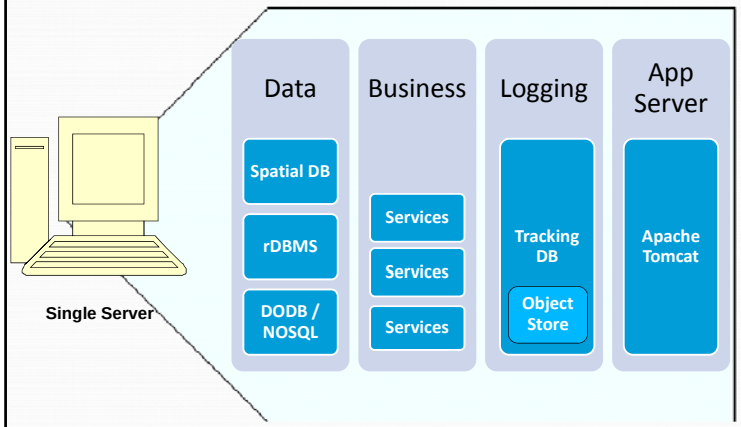
Wes J. Lloyd
August 7, 2012

Colorado State University, Fort Collins, Colorado USA

Outline

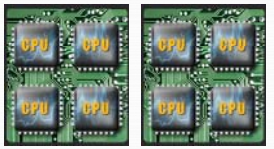
- Research Problem
- Challenges
- Approaches & Gaps
- Research Goals
- Research Questions & Experiments
- Research Contributions
- Preliminary Results

Traditional Application Deployment



Infrastructure as a Service (IaaS) Cloud Computing

- Server partitioning of multi-core servers
- Hardware virtualization
- Service isolation
- Resource elasticity

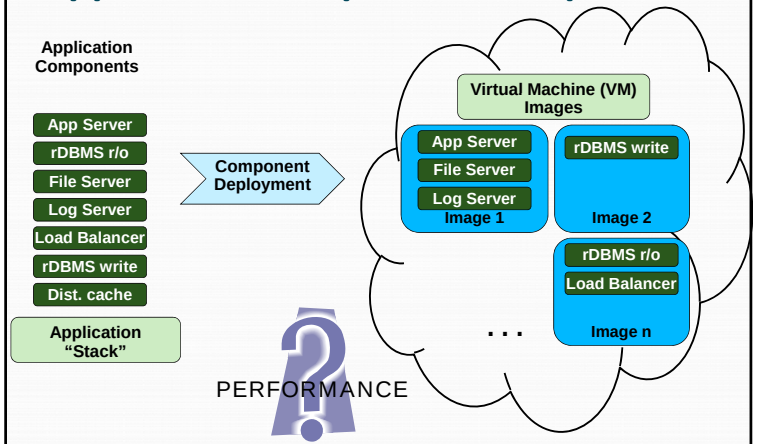


Problem Statement

- Autonomic deployment of multi-tier applications to IaaS clouds
  - Component composition
    - Collocation and interference of components
  - Scaling infrastructure to meet demand

Challenges

Application Component Composition



Bell's Number

n = number of application components

Model
Database
File Server
Log Server

Application "Stack"

1 VM : 1..n components

Component Deployment

VM deployments

config 1 (M D F L)
config 2 (M | D F L)
config n (M L F | D)
...

n	k
4	15
5	52
6	203
7	877
8	4,140
9	21,147
n	...

# of Configurations

k= # of possible configs

Provisioning Variation

Request(s) to launch VMs

VM
VM VM
VM

Ambiguous Mapping

VMs Share PM CPU / Disk / Network

Physical Host
Physical Host
Physical Host
Physical Host
Physical Host
Physical Host

PERFORMANCE

VMs Reserve PM Memory Blocks

Infrastructure Management

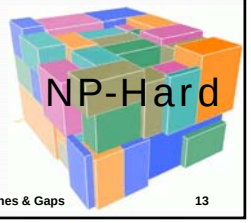
- Scale Services
- Tune Application Parameters
- Tune Virtualization Parameters

Service Requests
Load Balancer
Application Servers
Load Balancer
distributed cache
rDBMS
noSQL data stores

Approaches & Gaps

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

- Bins= physical machines (PMs)
- Items= virtual machines (VMs)
- Dimensions
  - CPU time
  - VM RAM, hard disk size, # cores
  - Disk read/write throughput
  - Network read/write throughput
- PM capacities vary dynamically
- VM resource utilization varies



Related Work

- Multivariate performance models
  - Regression models
  - Machine learning
- Feedback loop control
- Hybrid approaches
- Formal approaches
  - Integer linear programming
  - Case based reasoning

Gaps in Related Work

- Existing approaches do not consider
  - VM image composition
  - Complementary component placements
  - Interference among components
  - Minimization of resources (# VMs)
  - Load balancing of physical resources
- Performance models ignore
  - Disk I/O
  - Network I/O
  - VM and component location

Why Gaps Exist

- Public clouds
  - Research is cost prohibitive
  - Users concerned with performance not in control
- Private clouds: systems still evolving
- Performance models (large problem space)
- Virtualization misunderstood or overlooked

Research Goals

Research Goals

RG1: Support VM component composition

RG2: Support virtual infrastructure management

- Determine and execute VM placement
- Scale infrastructure for application demand

Performance Objectives

- Primary: Maximize application throughput
- Secondary: Minimize resource cost (# of VMs)
- Minimize modeling time
- Support high responsiveness to change in application demand

Research Questions & Methodology

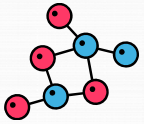
Methodology Evaluation

- CSIP: USDA-NRCS platform for model services
- Models as multi-tier application surrogates
  - RUSLE2 – Soil erosion model
  - WEPS – Wind Erosion Prediction System
  - Hydrology models: SWAT, AgES
  - Other models: STIR, SCI…
- Eucalyptus IaaS cloud(s)
  - Amazon EC2 compatible
  - XEN & KVM hypervisors

Component Composition	Infrastructure Management
RQ1	RQ3
	RQ4
RQ2	RQ5

RQ1: Which independent variables best help model application performance (throughput) to guide autonomic component composition?

- Total (all VMs) resource utilization
  - CPU time, disk I/O, network I/O, …
- Individual VM and PM resource utilization
- Component and VM location
- VM Configuration: number of cores, RAM, hypervisor type (KVM, XEN...)



RQ1: Which independent variables best help model application performance (throughput)

Methodology

Exploratory performance modeling

- Investigate independent variables
- Investigate modeling techniques
  - Multiple linear regression (MLR)
  - Artificial neural networks (ANNs)
  - Others

RQ2: Can component resource classifications and behavioral rules predict performance of component compositions?

- Support simplification of the search space
- Support applications with large # of components
  - Bell's number

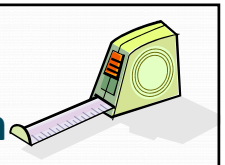
n	k
4	15
5	52
6	203
7	877
8	4,140
9	21,147
n	. . .

RQ2: Can component resource classifications and behavioral rules predict performance of

Methodology
Investigate autonomic component composition approach(es)

- Performance modeling
- Heuristics to classify
  - Component resource utilization
  - Component dependencies

Evaluation Metrics: Component Composition



- Composition performance
  - Average throughput of configurations
- Resource packing density
  - # components/# VMs for compositions
- Derivation speed
  - Average wall clock time to produce compositions

RQ3: Does performance of component compositions change when scaled up?

- Single provisioned application VMs → Multiply provisioned application VMs
- Investigate collocation of new VMs
  - Intelligent vs. ad-hoc placement
  - Load balance physical resources

RQ3: Do performance rankings of component compositions change when scaled up?

Methodology
Scale up compositions and benchmark performance

•Investigate impact of VM placement for infrastructure scaling

RQ4: How rapidly can VMs be launched in response to application demand?

- Determine upper bound of VM launch speed
- Devise workarounds to improve performance
  - VM prelaunch and suspension
    - Reserve RAM, other resources multiplexed
  - Enforced caching of VM data on PMs
  - Reassign duties of existing VMs
  - Others ?

RQ4: How rapidly can VMs be launched in response to application demand?

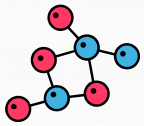
- Determine upper bound of VM launch speed

Methodology
Benchmark VM launch performance and investigate potential improvements

- Reassign duties of existing VMs
- Others ?

RQ5: Which independent variables best support application performance modeling for autonomic infrastructure management?

- Virtual infrastructure
  - Number of VMs (1 to n) per application VM
  - VM RAM, # cores
  - VM location data
- Application specific parameters
  - Number of worker threads
  - Number of database connections
  - Number of app server concurrent connections



RQ5: Which independent variables best support application performance modeling for autonomic infrastructure management?

Methodology

Explore autonomic infrastructure management approach(es)

- Performance model based
- Feedback control
- Hybrid approach

- Number of database connections
- Number of app server concurrent connections

Evaluation Metrics: Infrastructure Management

- Percentage of service requests completed
- Responsiveness
  - Max supported load acceleration without dropping requests
- Adaptation time
  - Time window with dropped requests
- Failure recovery time

Contributions

Expected Contributions (1/2)

- Novel, intelligent approaches for IaaS cloud
  - Application deployment
  - Infrastructure management
- Move IaaS infrastructure management beyond simple management of VM pools

Expected Contributions (2/2)

- Autonomic component composition (RQ1, RQ2)
- Autonomic infrastructure management (RQ3, RQ4, RQ5)
- Improve application performance modeling
  - For component composition (RQ1)
  - New independent variables (RQ1)
  - Heuristics (RQ2)
  - For infrastructure management (RQ5)
- Support load balancing of physical resources

Non-goals

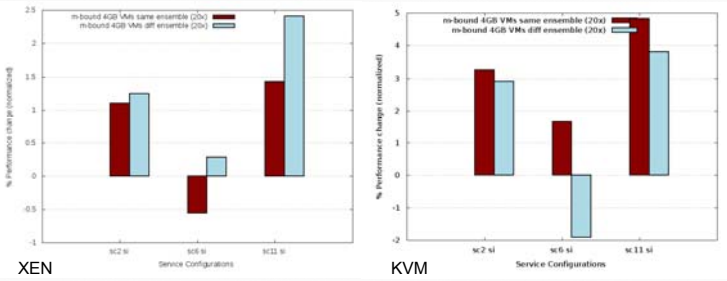
- Support for stochastic applications
  - Only applications with stable resource utilization characteristics supported
- External interference
  - From non-application VMs
- Hot-spot detection

Preliminary Results

Component Compositions

Rusle 2 Model Fastest Compositions

Service Isolation Overhead



XEN KVM

Component Compositions

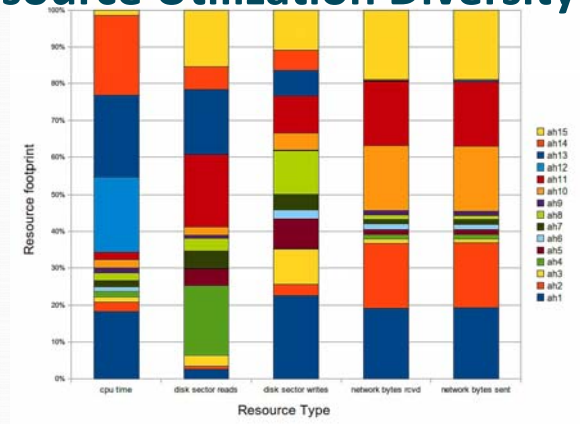
Rusle 2 Model Fastest Compositions

SC2 SC6 SC11

- Determining fastest compositions
  - Not intuitive
  - Testing / prediction required
- Service Isolation
  - Was not fastest
  - Adds overhead
  - Results in maximum hosting costs (# of VMs)

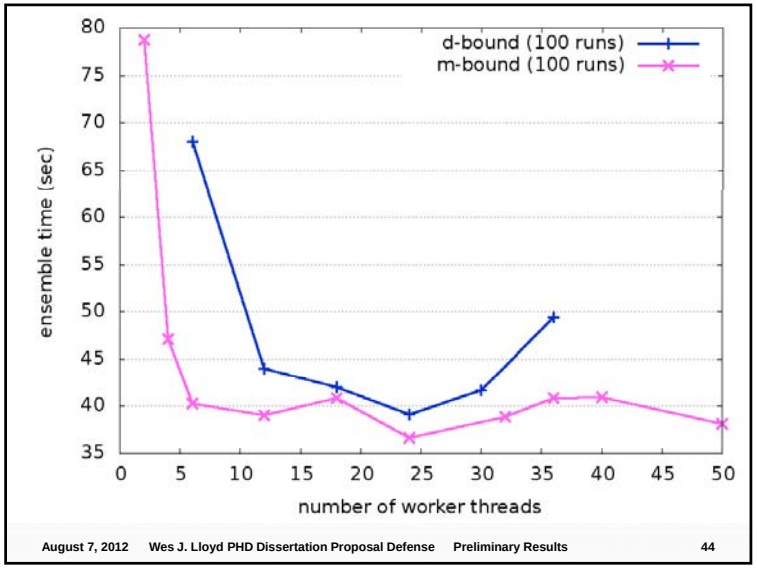
XEN KVM

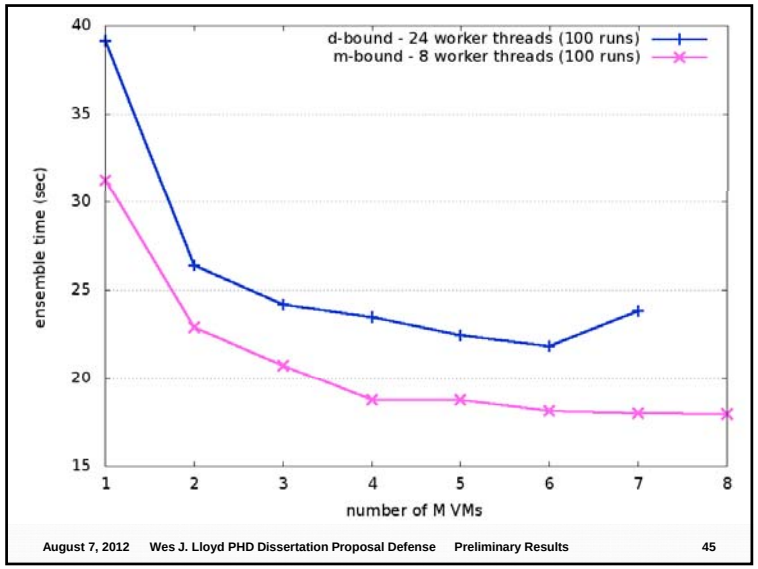
Component Composition Resource Utilization Diversity

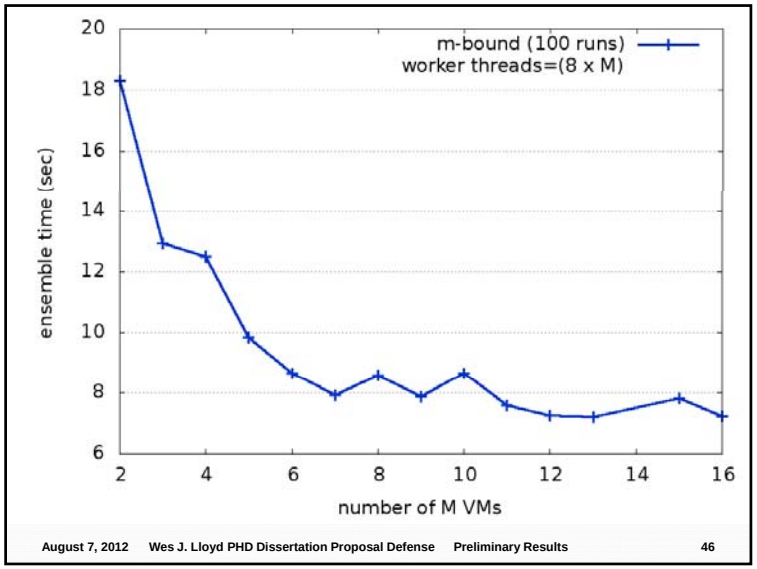


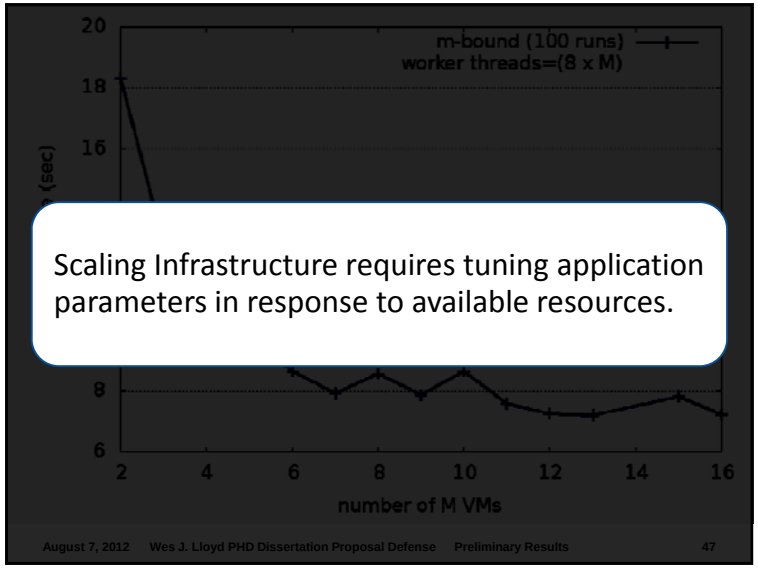
Component Composition Resource Utilization Diversity

- Resource utilization varies
  - Component/VM placements
  - VM memory size allocations
  - Hypervisor type KVM/XEN
- Testing required to identify resource utilization
- Intuition is insufficient




d-bound - 24 worker threads (100 runs)
m-bound - 8 worker threads (100 runs)
ensemble time (sec)
number of M VMs
40
35
30
25
20
15
1
2
3
4
5
6
7
8


m-bound (100 runs)
worker threads=(8 x M)
ensemble time (sec)
number of M VMs
20
18
16
14
12
10
8
6
2
4
6
8
10
12
14
16


m-bound (100 runs)
worker threads=(8 x M)
(sec)
number of M VMs
20
18
16
8
6
2
4
6
8
10
12
14
16
Scaling Infrastructure requires tuning application parameters in response to available resources.


Questions