JustRunIt: Experiment-Based Management of Virtualized Data Centers

Wei Zheng
Ricardo Bianchini

Rutgers University

Yoshio Turner
Renato Santos
John Janakiraman

HP Labs

Motivation

- Managing data center is a challenging task
  - Resource allocation, evaluation of software/hardware upgrades, capacity planning, etc.
  - Decisions affect performance, availability, energy consumption
- State-of-the-art uses modeling for these tasks
  - Models give insight into system behavior
  - Fast exploration of large parameter spaces
- Modeling has some important drawbacks
  - Consumes a very expensive resource: human labor
  - Needs to be re-calibrated and re-validated as the systems evolve

Our Approach

- Idea: experiments are a better approach
  - Consume a cheaper resource: machine time (and energy)
  - High fidelity
- JustRunIt: an infrastructure for experiment-based management of virtualized data centers
- Management system or administrator can use JustRunIt results to perform management tasks
  - Resource management and hardware/software upgrades
  - Select the best value for software tunables
  - Evaluate the correctness of administrator actions

Outline

- Motivation
- JustRunIt design and implementation
- Evaluation
  - Case study 1: resource management
  - Case study 2: hardware upgrades
- Related work
- Conclusion

Target Environment

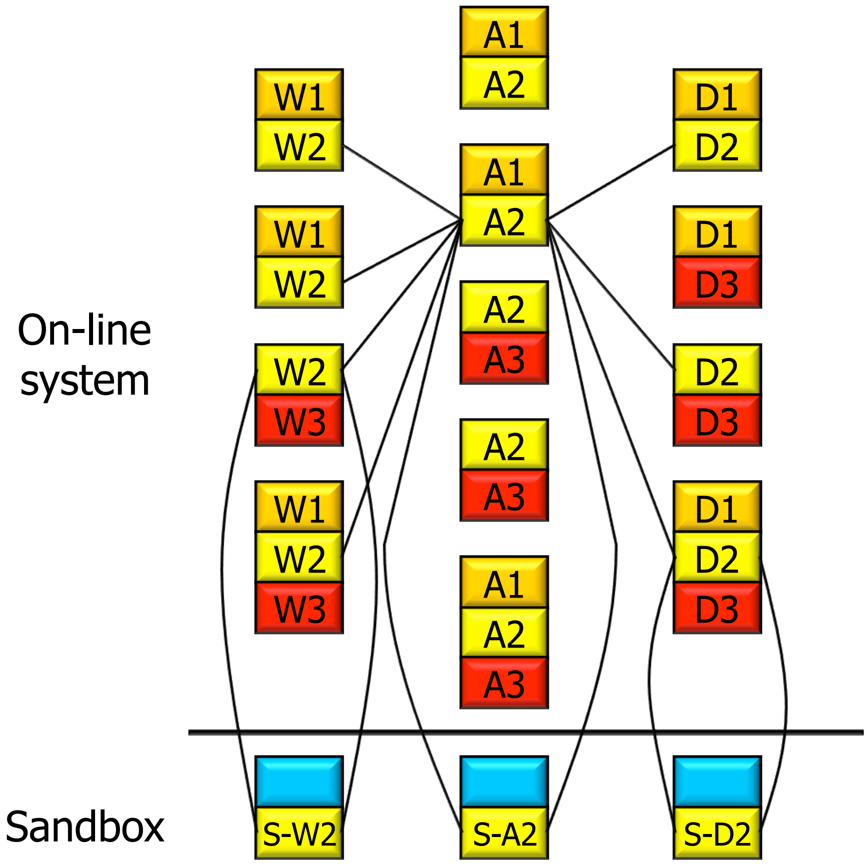
- Virtualized data centers host multiple independent Internet services

- Each service comprises multiple tiers, e.g. a web tier, an application tier, and a database tier

- Each service has strict negotiated SLAs (Service Level Agreements), e.g. response time

- All services are hosted in VMs for isolation, easy migration, management flexibility

Data Center with JustRunIt

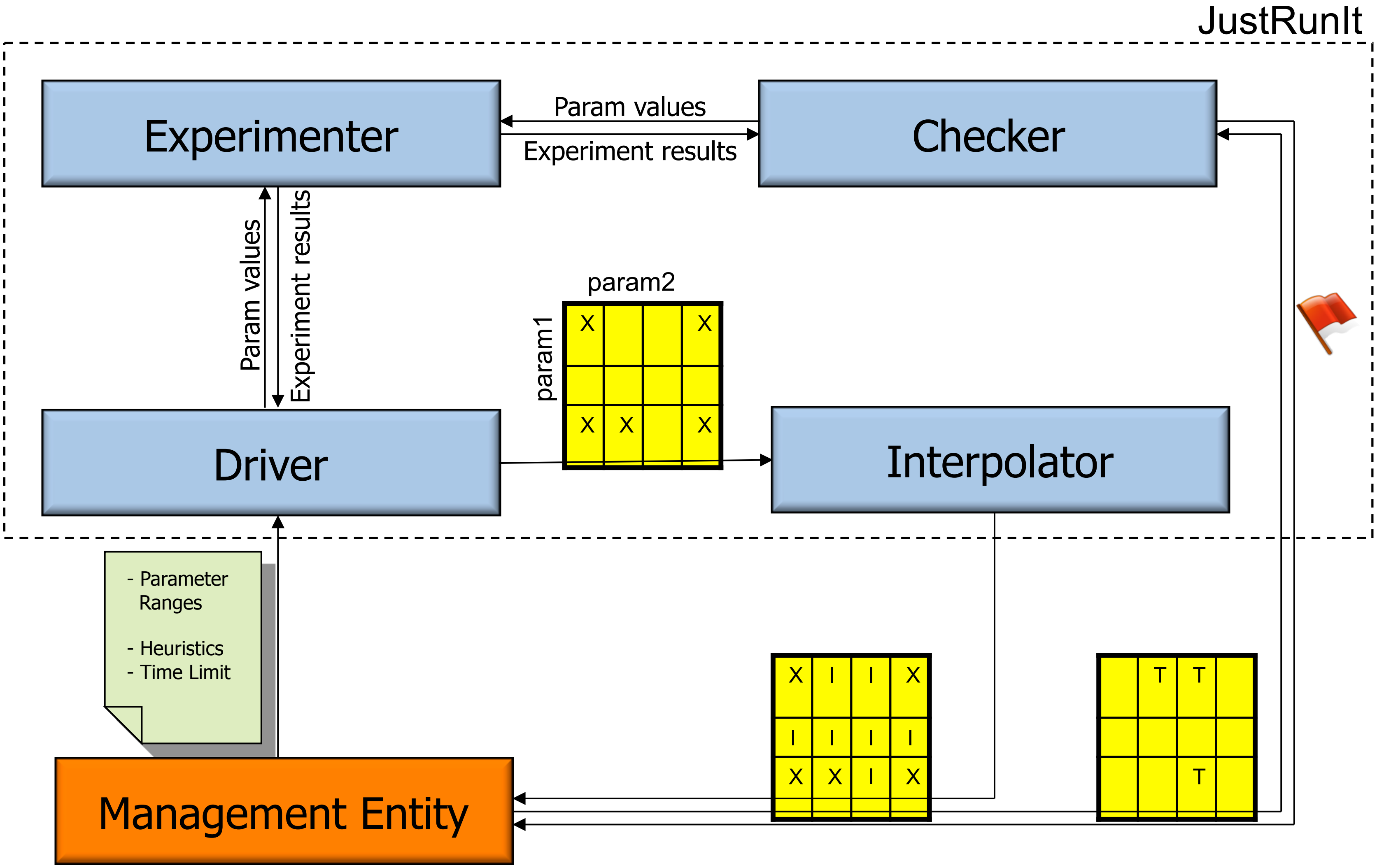


Assess performance and energy of different configurations

- Creates sandbox
- Clones VMs
- Applies configuration changes
- Duplicates live workload to sandbox

- Properties
  - No effect on on-line services
  - Does not replicate entire service
  - Almost service-independent

JustRunIt Architecture

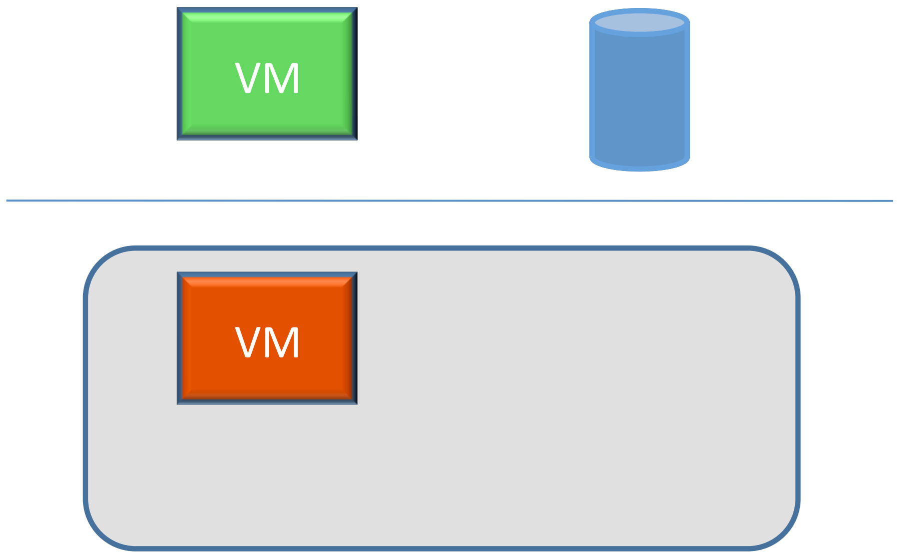


X			X
X	X		X

X	I	I	X
I	I	I	I
X	X	I	X

	T	T	
		T	

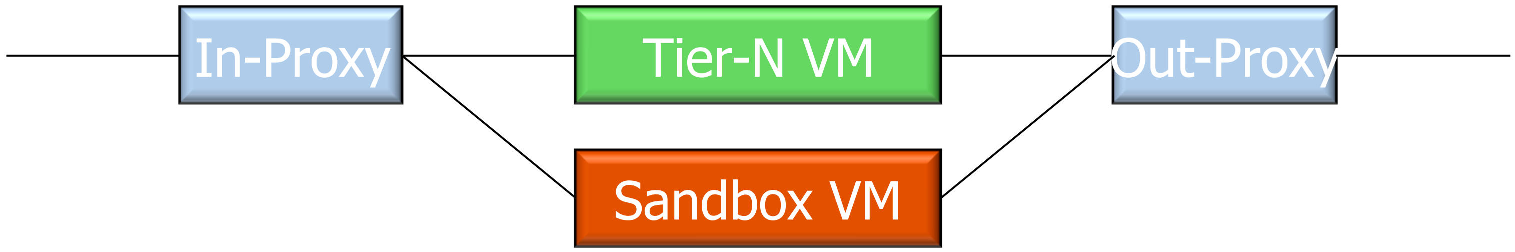
Experimenter



- Step 1: Clone subset of production system to a sandbox
  - VM cloning: Modify Xen live migration to resume original VM instead of destroying it
  - Storage cloning: LVM copy-on-write snapshot for sandbox VM
  - L2/L3 network address translation: implemented in driver domain netback driver to prevent network address conflict
- Step 2: Apply configuration changes
  - Exs: CPU allocation, frequency

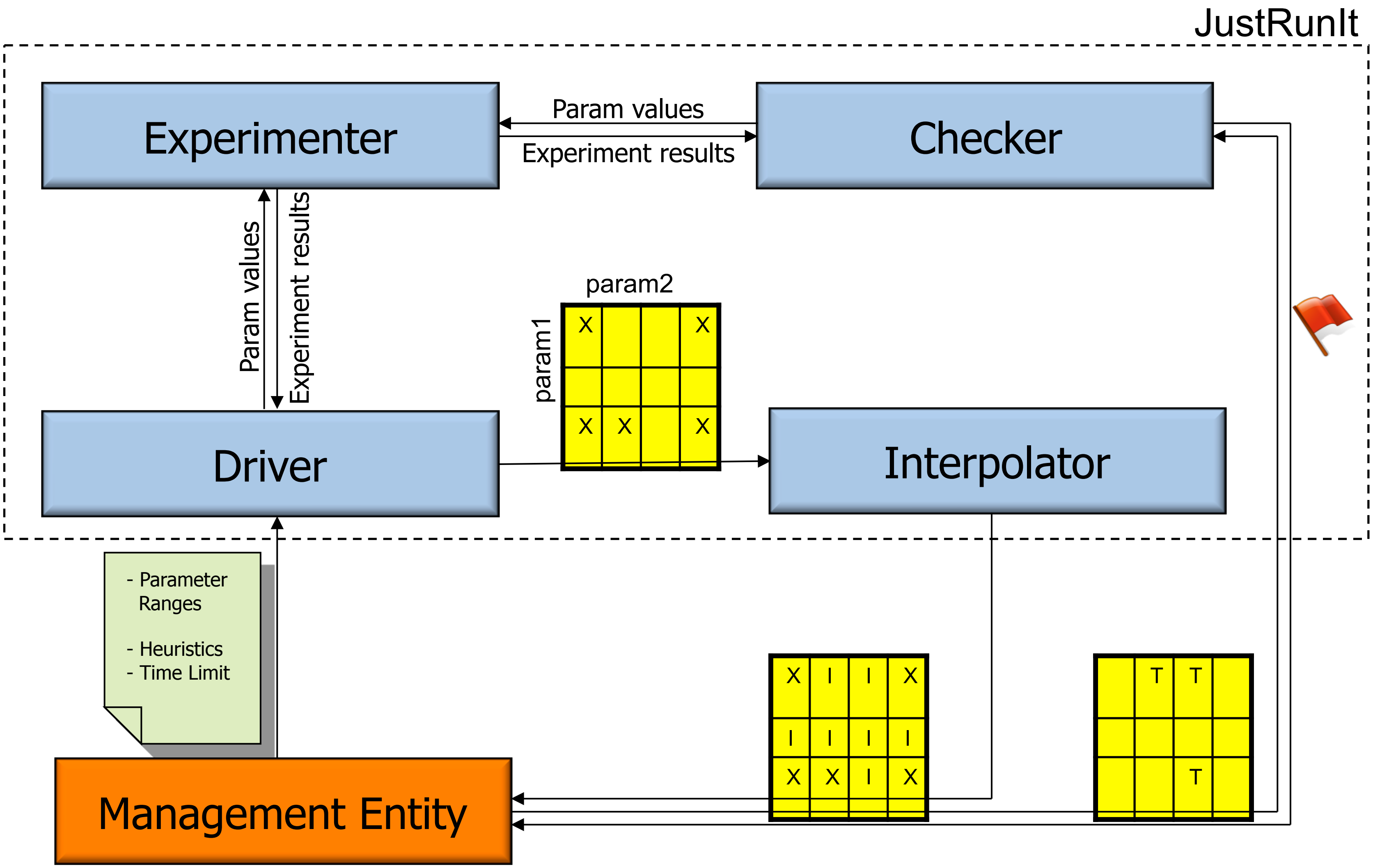
Experimenter

- Step 3: Duplicates live workload to sandbox using proxies



- Proxies filter requests/replies from the sandbox VM
- Emulates the timing and functional behavior of preceding and following service tiers
  - Application protocol level requests/replies (e.g. HTTP)

JustRunIt Architecture



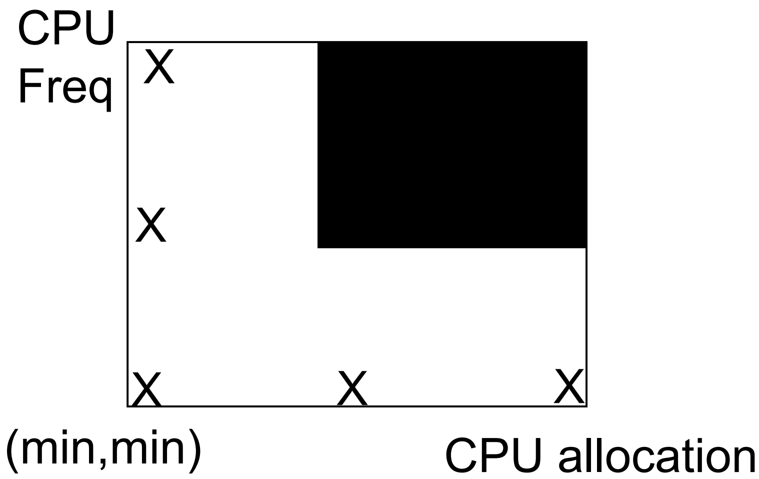
X			X
X	X		X

X	I	I	X
I	I	I	I
X	X	I	X

	T	T	
		T	

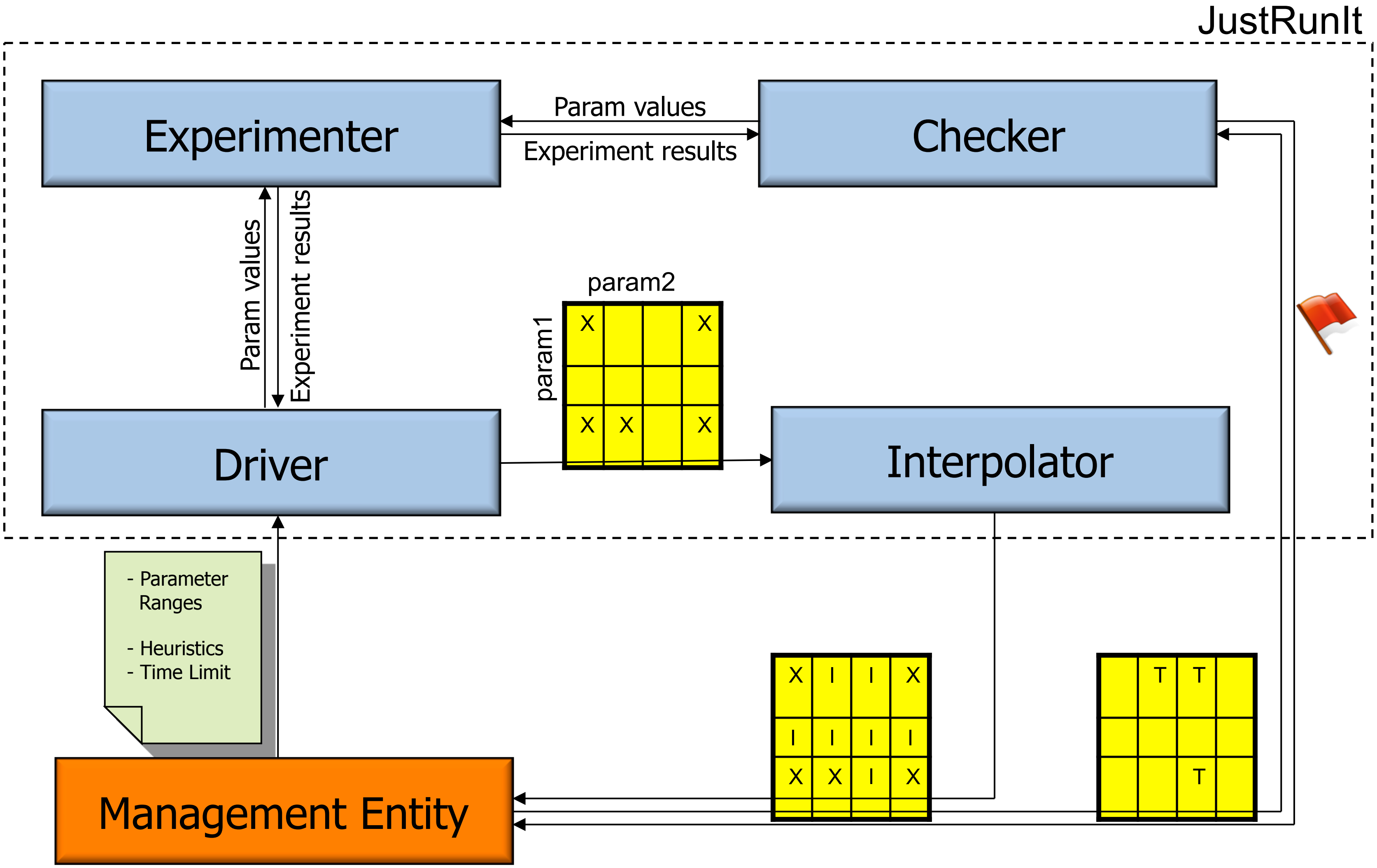
Driver

- Goal: Fill in results matrix within a time limit



- Corners
- Midpoints (recursive)
- Heuristics
  - Cancel experiments if gain for a resource addition falls below a threshold
  - Cancel experiments for tiers that do not produce the largest gains from a resource addition

JustRunIt Architecture



Interpolator and Checker

- For simplicity, we use linear interpolation
- Checker will verify the interpolated result by invoking the experimenter to run corresponding experiments in the background

Cost of JustRunIt

- Building JustRunIt needs human effort also
  - The most time-consuming part is proxies implementation
  - Current proxies understand HTTP, mod_jk, MySQL protocols
  - Developed from an open source proxy daemon, each proxy need 800~1500 new lines of C code

- Cost of VM Cloning: 42 lines of Python code in xend and 244 lines of C in netback driver

- The engineering cost of JustRunIt can be amortized for any service based on the same protocols

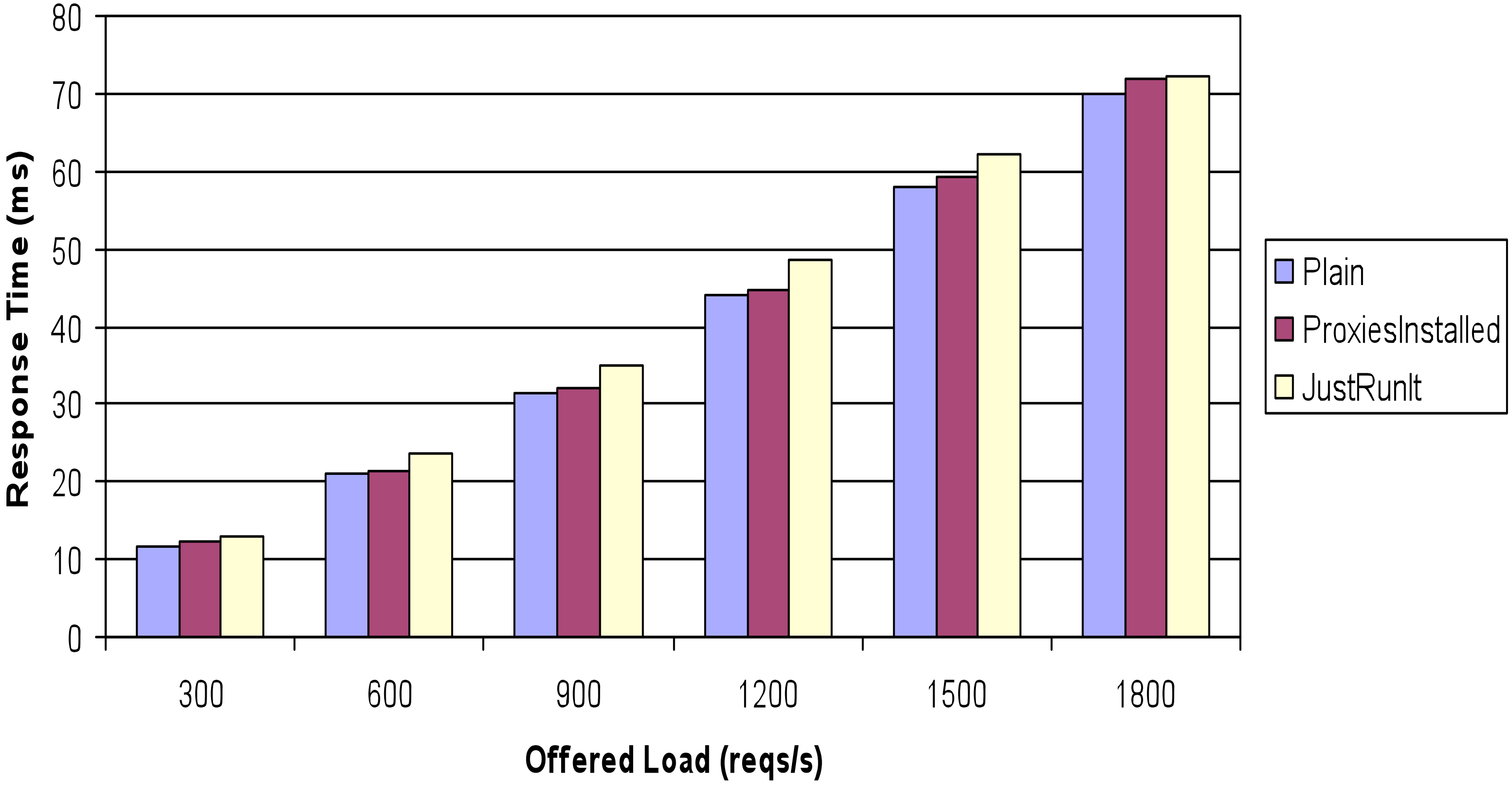
Outline

- Motivation
- JustRunIt design and implementation
- Evaluation
  - Case study 1: resource management
  - Case study 2: hardware upgrades
- Related work
- Conclusion

Methodology

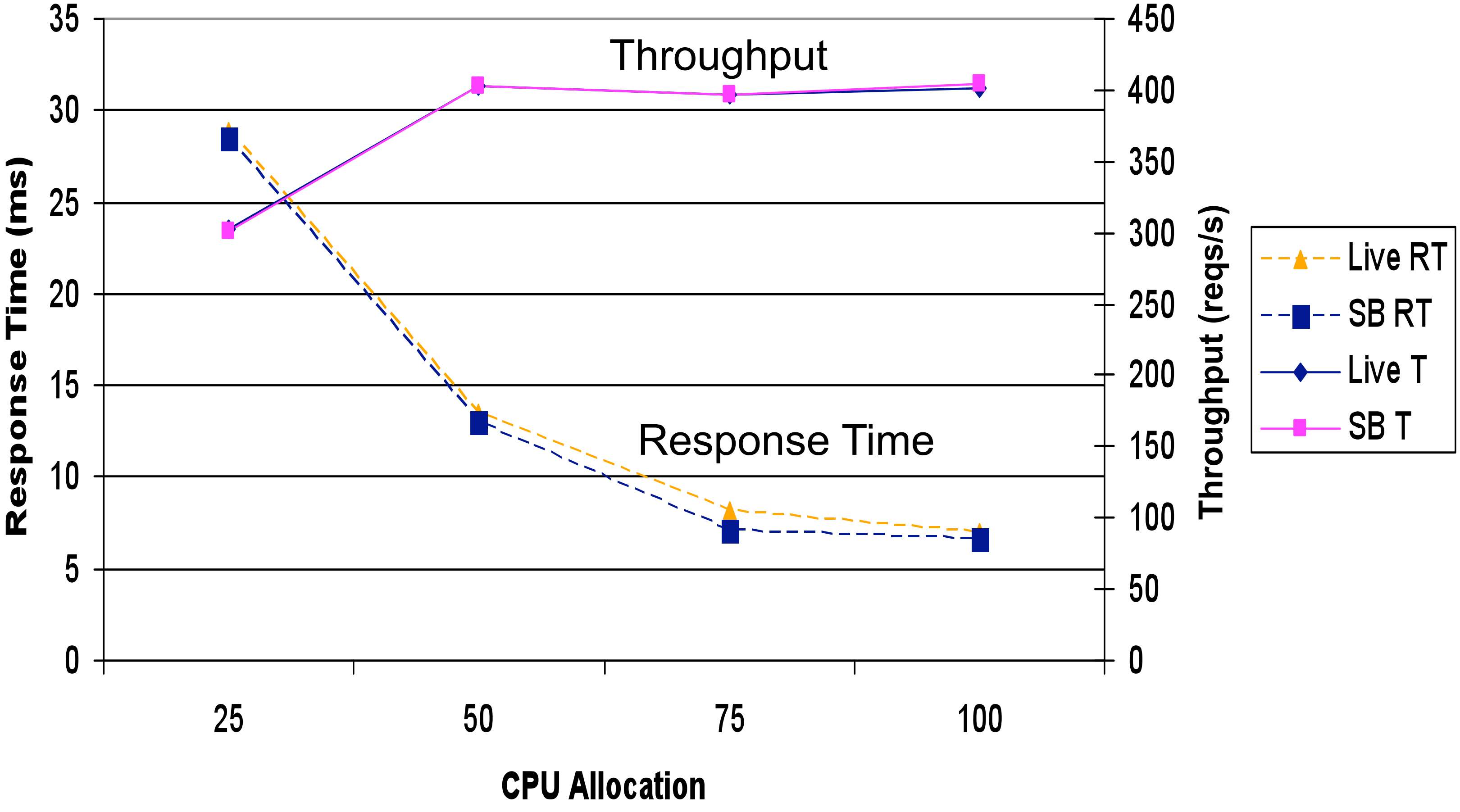
- 15 HP Proliant C-class blades (8G, 2 Xeon dual-core) interconnected with Gbit network
- 2 types of 3 tier Internet service
  - RUBiS: online auction service modeled after Ebay.com
  - TPC-W: online book store modeled after Amazon.com
- Xen 3.3 with Linux 2.6.18
- Dom0 pinned to separate core for performance isolation

Overhead on On-line Service?



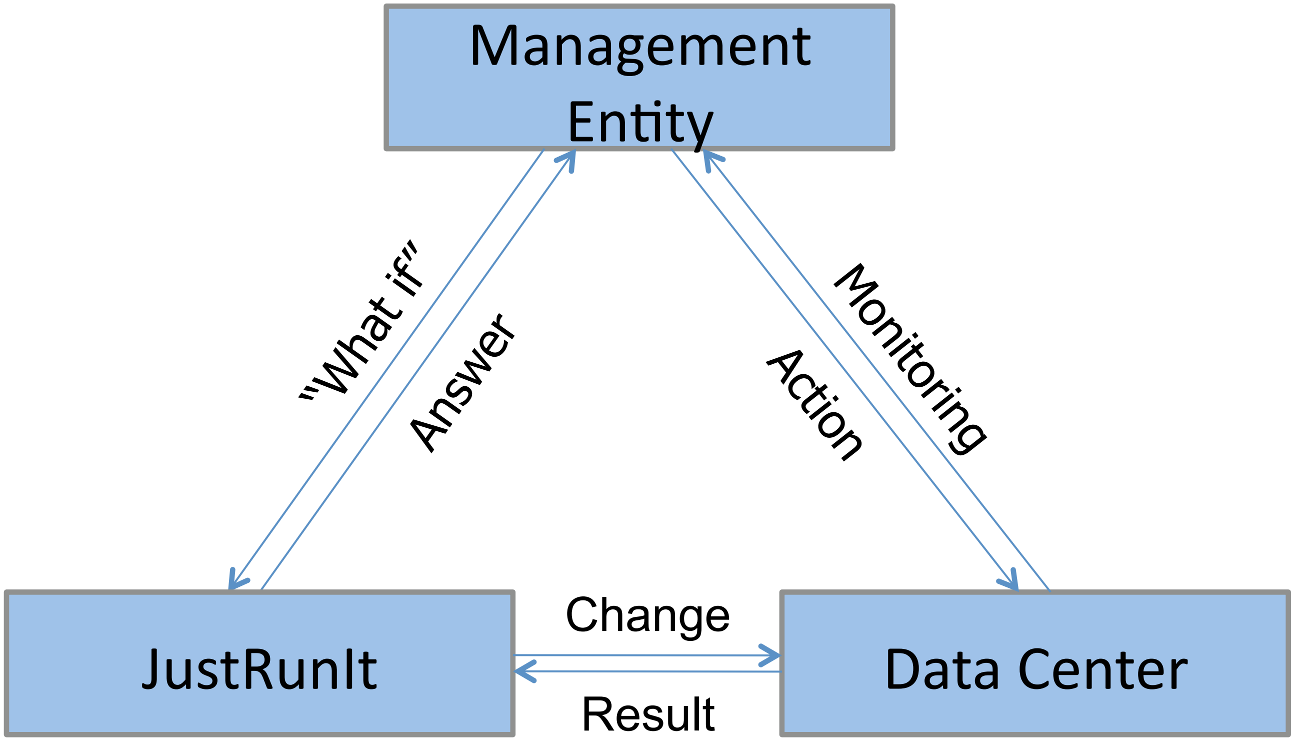
3-tier service with one node per tier; two nodes for proxies
Overhead exposed – slight RT degradation, no effect on TP

Fidelity of The Sandbox Execution?



Application server at 400 requests/second (similar results for higher load)

Automated Management



Management Entity

"What if"

Answer

Action

Monitoring

JustRunIt

Change

Result

Data Center

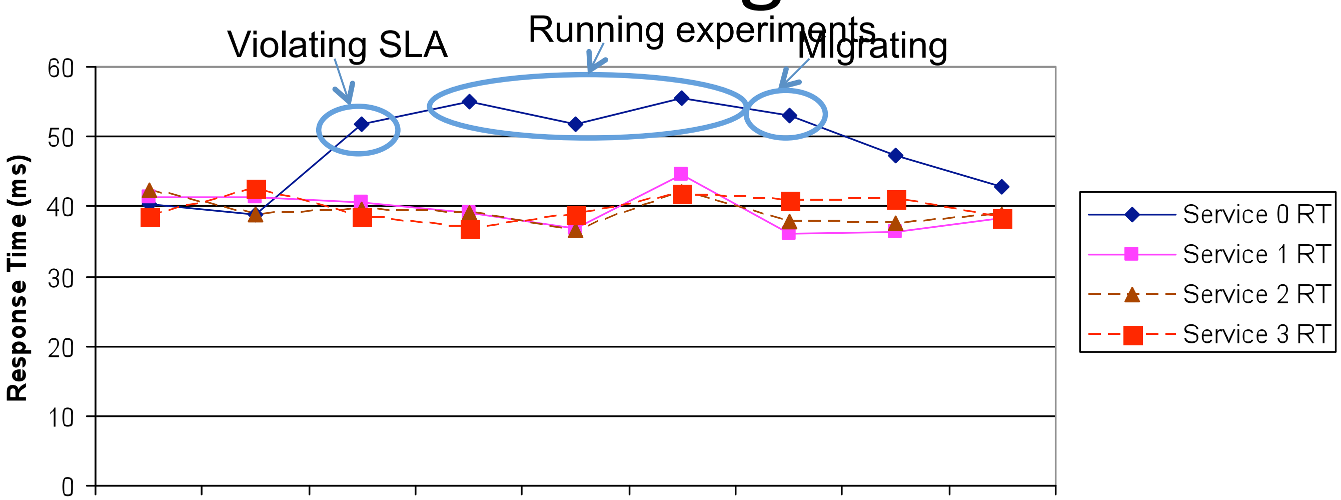
Case Study 1: Resource Management

- Goal: consolidate the hosted services onto the smallest possible set of nodes, while satisfying all SLAs
- Management entity invokes JustRunIt when response time SLA is violated, or when SLA is met by a large margin
- Management entity uses performance-resource matrix to determine resource needs
- Management entity performs bin packing (via simulated annealing) to minimize number of physical machines and number of VM migrations

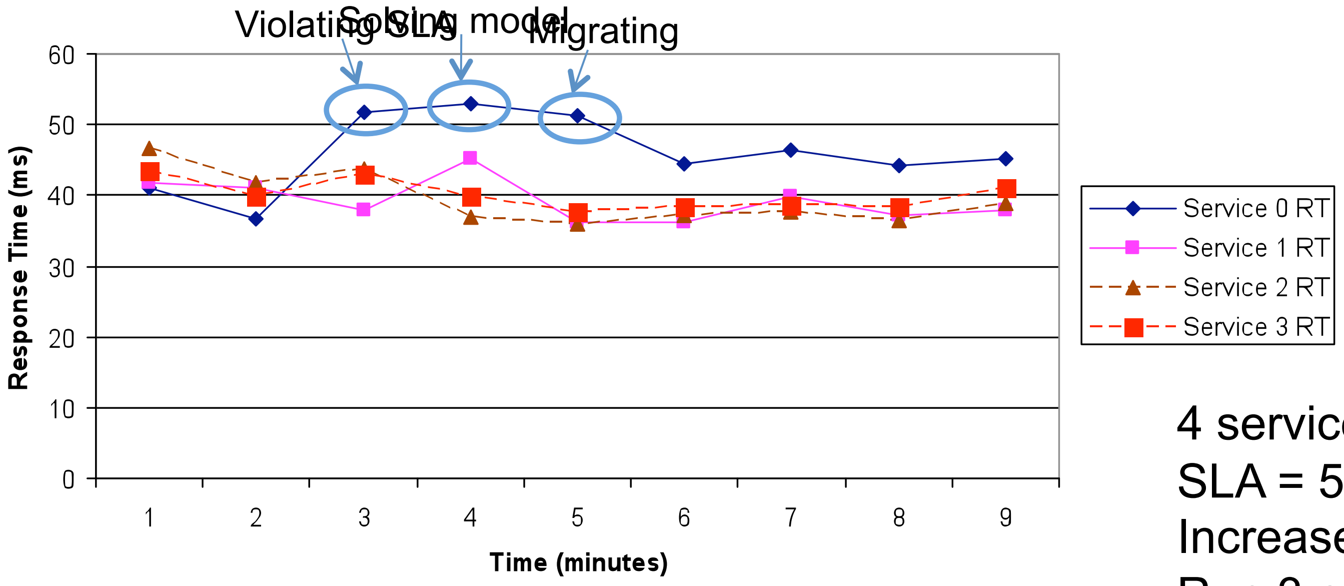
Case Study 1: Resource Management

- 9 blades: 2 for first tier; 2 for second tier; 2 for third tier; 3 for load balancing and storage service
- 4 services are populated
- Each VM allocated 50% CPU
- SLA: 50ms
- Service 0 workload is increased to 1500reqs/sec after 2 mins

Resource Management with JustRunIt



JRI



Modeling

4 services on 11 nodes
SLA = 50ms
Increase load on S0
Run 3 exps for 3 mins

Case Study 2: Hardware Upgrades

- Goal: evaluate if hardware upgrade allow further consolidation and lower overall power consumption

- JustRunIt uses one instance of new hardware in sandbox to determine the consolidation savings

- Bin packing determines necessary number of new machines to accommodate production workload

Case Study 2: Hardware Upgrades

- Initial server uses 90% of one CPU core on old hardware (emulate using low frequency mode)
- New machine (emulate using high frequency mode) requires 72%
- This would allow further consolidation in a large system

Related Work

- Modeling, feedback control, and machine learning for managing data centers [Stewart'05, Stewart'08, Padala'07, Padala'09, Cohen'04]

- Scaling down data centers emulation [Gupta'06, Gupta'08]

- Sandboxing and duplication for managing data centers [Nagaraja'04, Tan'05, Oliveira'06]

- Run experiments quickly [Osogami'06, Osogami'07]

- Selecting experiments to run [Zheng'07, Shivam'08]

Conclusions

- JustRunIt infrastructure combines well with automated management systems
- Answers “what-if” questions realistically and transparently
- Can support a variety of management tasks
- Future investigation
  - Tier interactions
  - Different workload mix
  - Build proxies for a database server

THANK YOU!

QUESTIONS?