DAPA: Diagnosing Application Performance Anomalies for Virtualized Infrastructures

Hui Kang, Stony Brook University

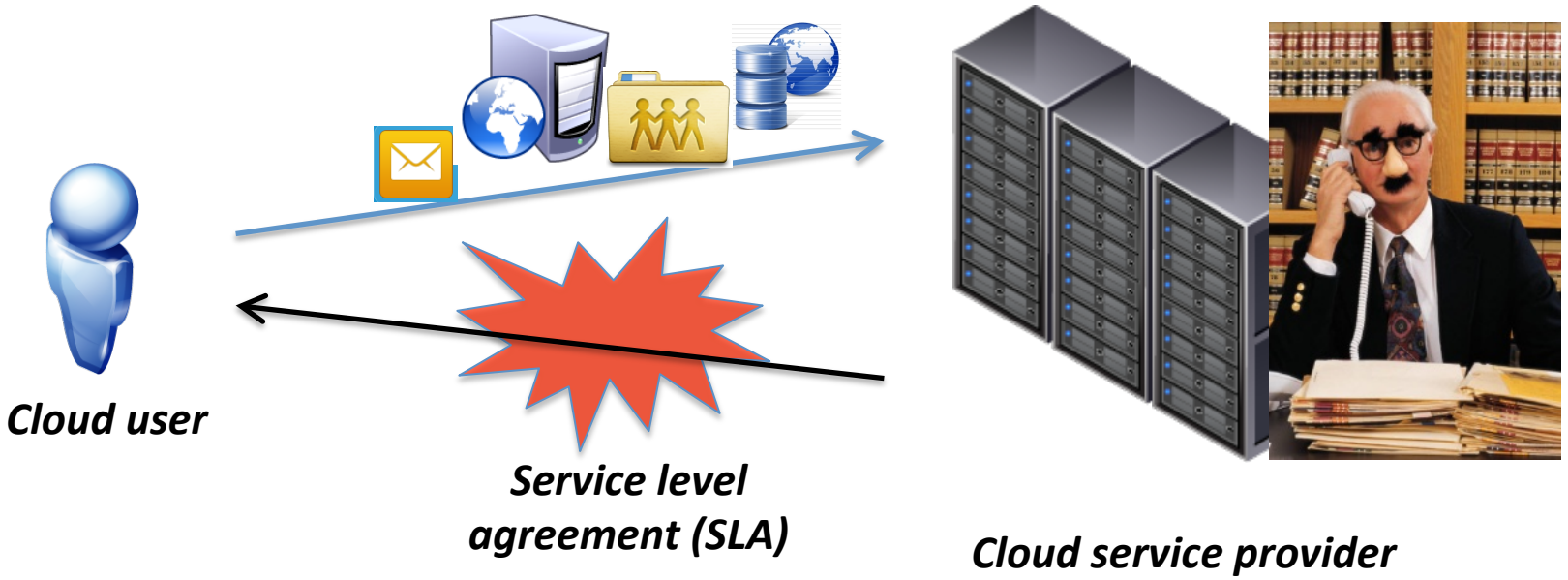
Xiaoyun Zhu, VMware

Jennifer Wong, Stony Brook University



vmware®

Performance management in cloud computing



- Actions taken by the system administrator
  - Localize the faulty application or system components
  - Find the root cause
  - Bring the system back to normal status

Existing solutions

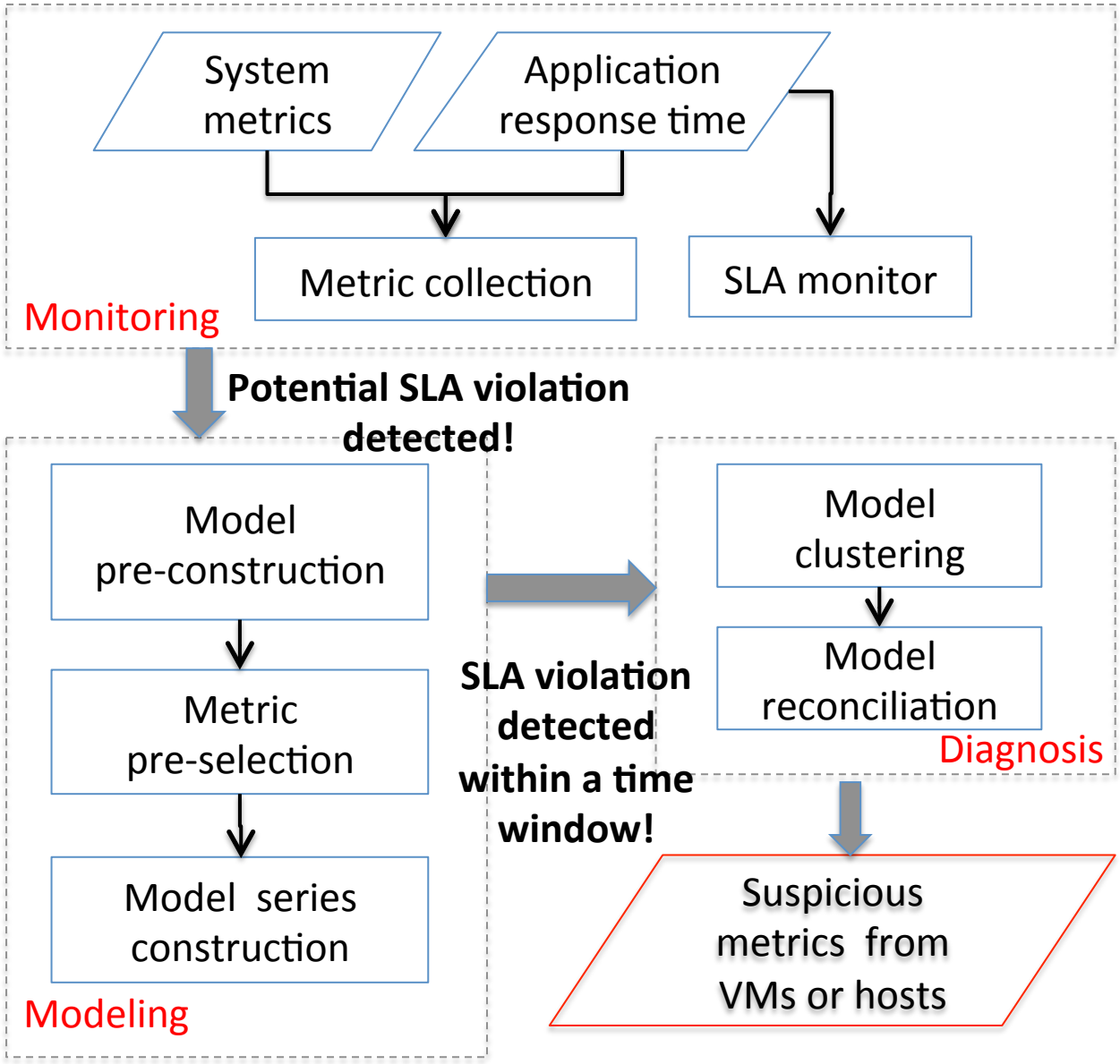
- Domain knowledge in specific application/platforms and associated performance best practices, e.g., [B2007] (*labor intensive, non-scalable*)
- Dependency model approach , e.g, [B2003, K2005]
  - *heavy instrumentation*
  - *packet trace on each node*
- ML techniques modeling the relationship in distributed systems, [C2004] (*complexity high, unfit for online analysis*)
- Analytic performance models for adaptive resource provisioning of VMs, e.g., [P2009] (*model inputs determined by human experts, redundant and non-scalable*)

Mostly designed in the **Absence of Virtualization**

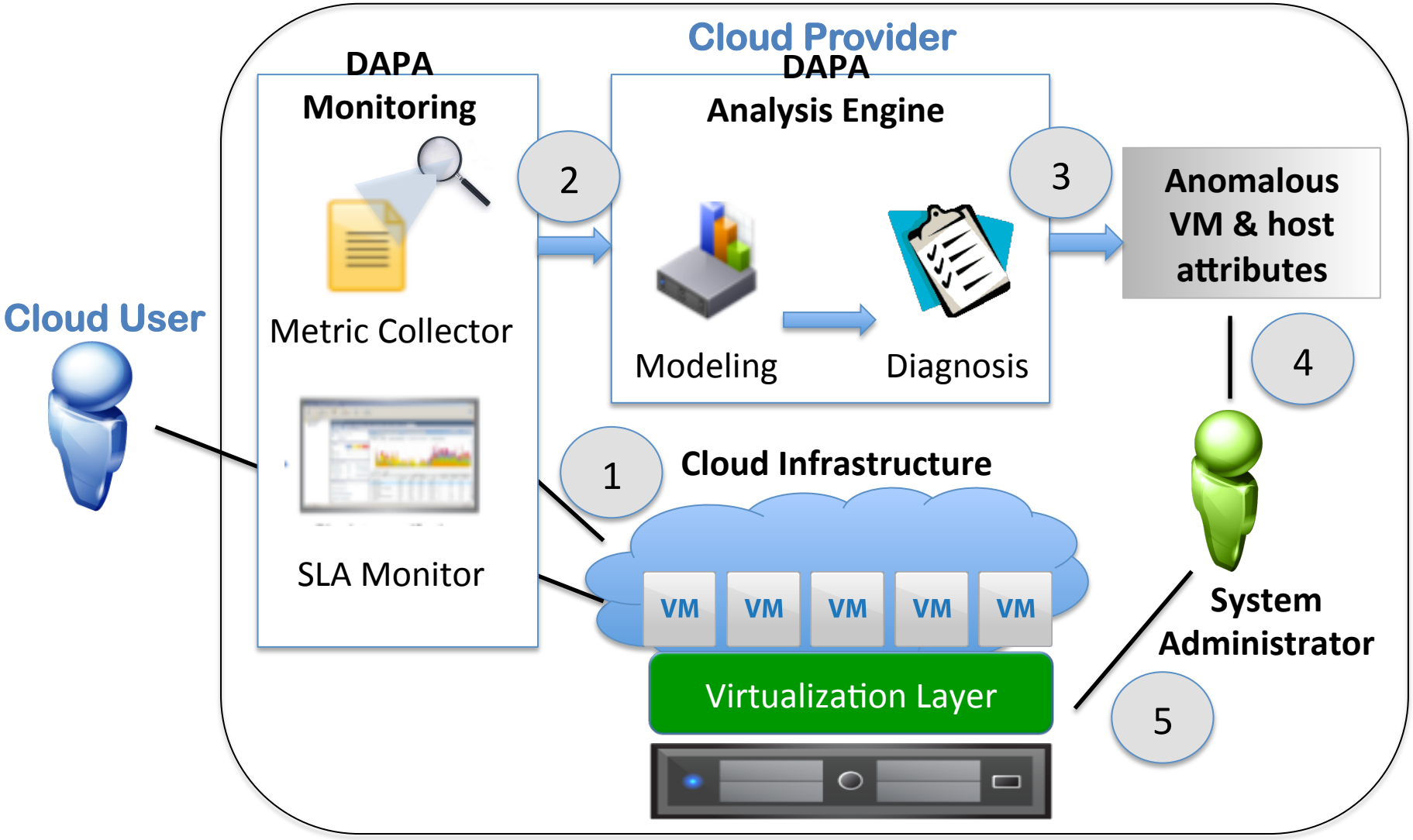
Characteristics of virtualized infrastructures

- Performance loss caused by resource contention between VMs in the same host
- Fine-grained, system-level performance counters
- VM migration
- Strong dependency between VMs and their hosts => strong correlated metrics
  - negative modeling accuracy
  - wasting computing time

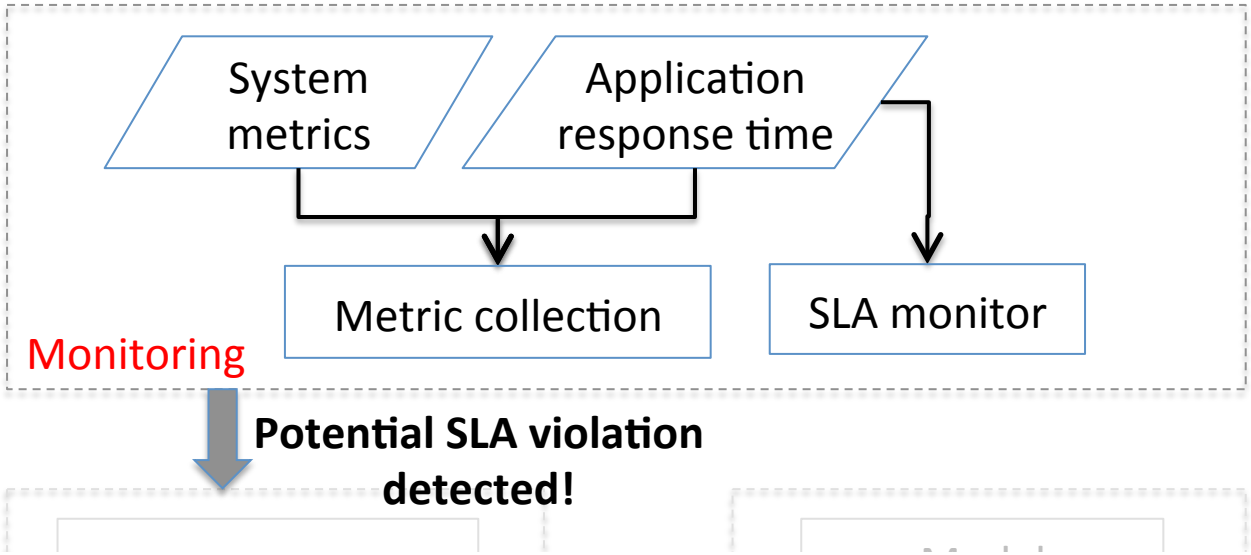
DAPA – *D*iagnosis *A*nd *P*erformance *A*nalysis



DAPA typical deployment



Online monitoring



- Collection application performance and system metrics periodically
  - Transaction response time
  - cpu utilization, memory usage, etc
- Raise an alarm on a *potential* SLA violation

Modeling

VMs or hosts

Potential SLA violation

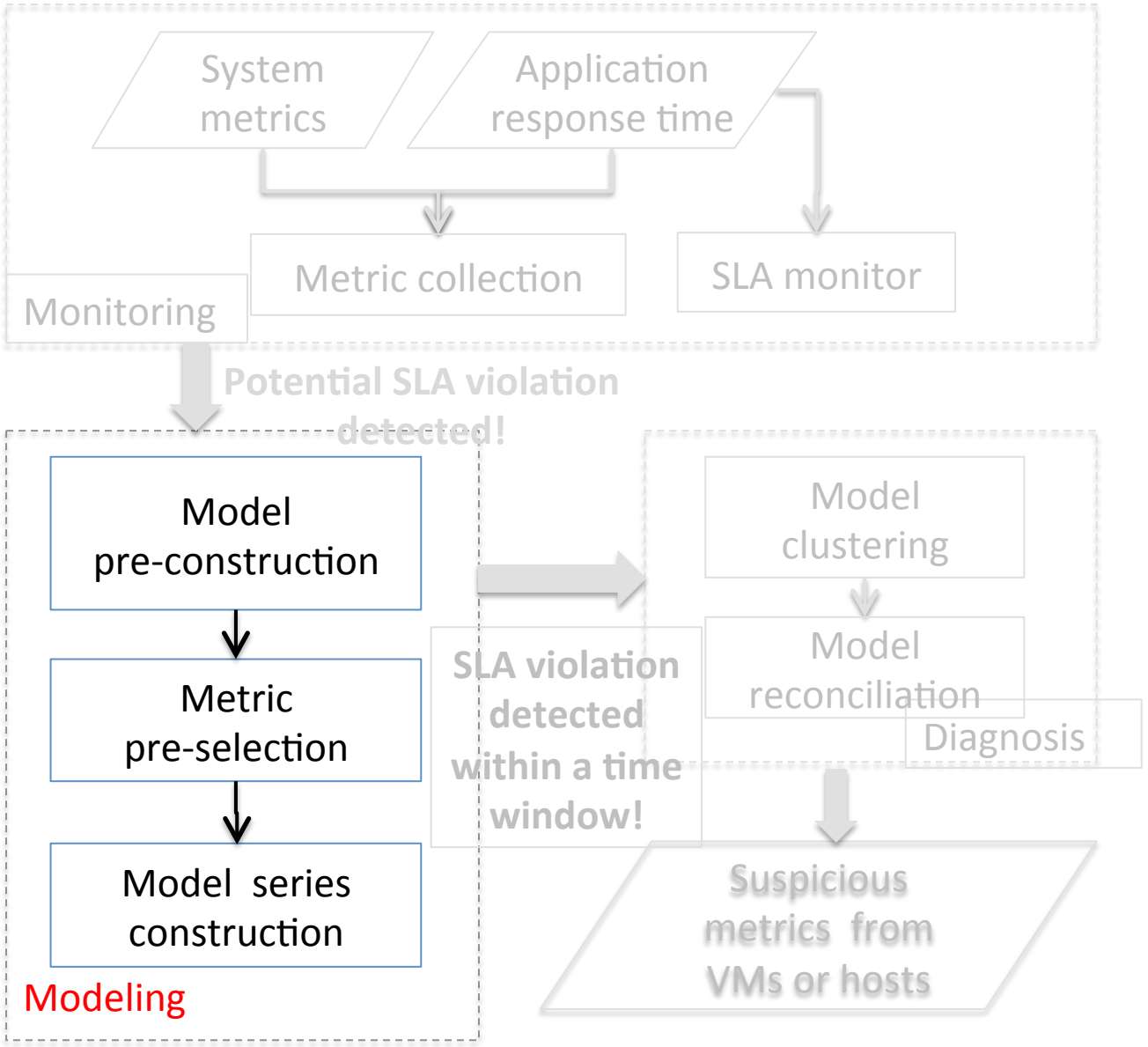
- A more stringent condition than true SLA violation
- Always occurs prior to a true SLA violation
- Trigger of the modeling phase

Example

- SLA violation: "*more than 10% of the response time for a certain type of transaction within one minute are greater than one second*".
- Potential SLA violation: *At least 5% are greater than one second*"

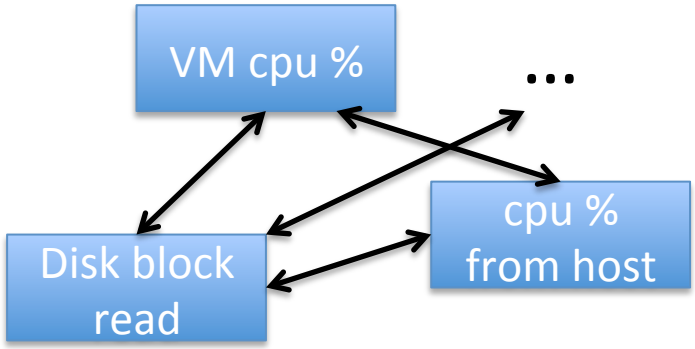
- Benefits – capturing the characteristics of the system in both normal and abnormal status

Modeling


System metrics
Application response time
Metric collection
SLA monitor
Monitoring
Potential SLA violation detected!
Model pre-construction
Metric pre-selection
Model series construction
Modeling
SLA violation detected within a time window!
Model clustering
Model reconciliation
Diagnosis
Suspicious metrics from VMs or hosts

Challenges

- Problems:
  - Lots of metrics and redundancy due to virtualization
  - Sensitivity of the models to data variations
  - Model overtuned



More than *hundreds of correlation pairs*
Brute force calculation is infeasible

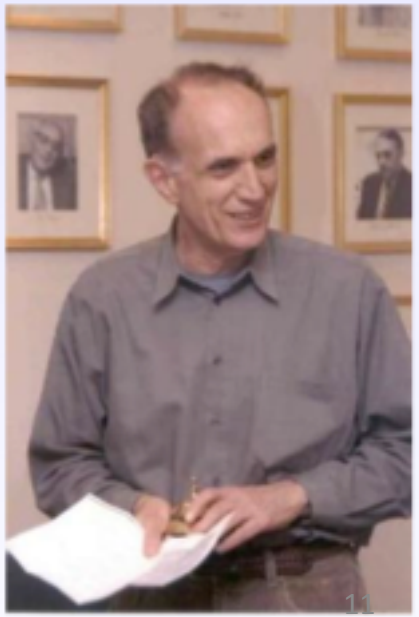
- Approach: need an appropriate modeling technique

Statistics

- Least angle regression (LARS) [E2004]
  - *Input*: $X_1, X_2, \ldots X_m$ – predicator variables and Y – response variable. All are *n*-dim vectors
  - *Output*: A set of vector of regression coefficients β that minimize the RSS, subject to a constraint on the sum of β.

$$\min \| \mathbf{Y} - \hat{\mathbf{Y}} \| = \sum_{j=1}^{n} \left(y_j - \sum_{i=1}^{m} \beta_i X_{ji} \right)^2 , s.t. \sum_{i=1}^{m} |\beta_i| \leq t.$$

RSS L_1 penalty

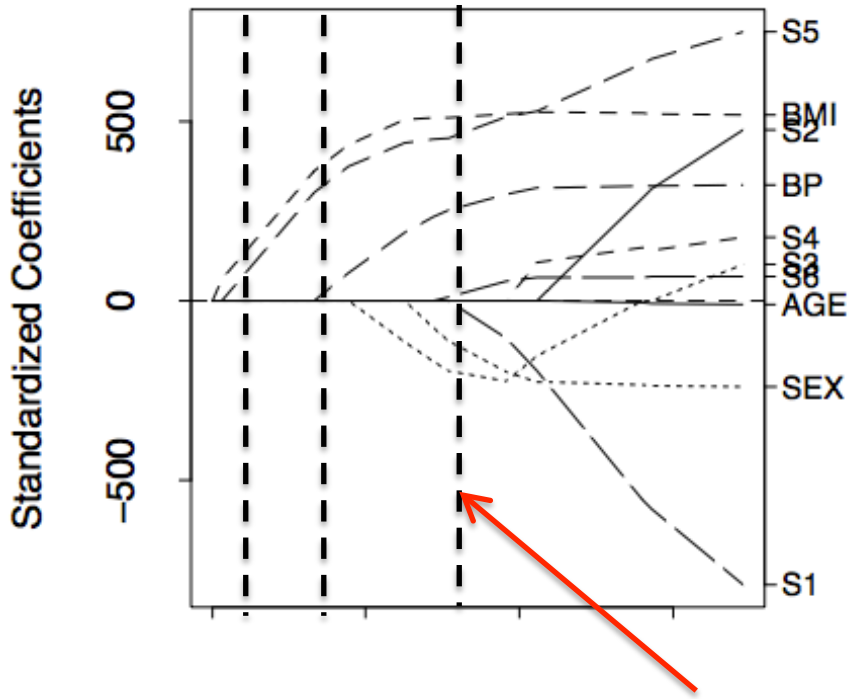


Prof. Bradley Efron
(Main inventor of LARS)

LARS regression

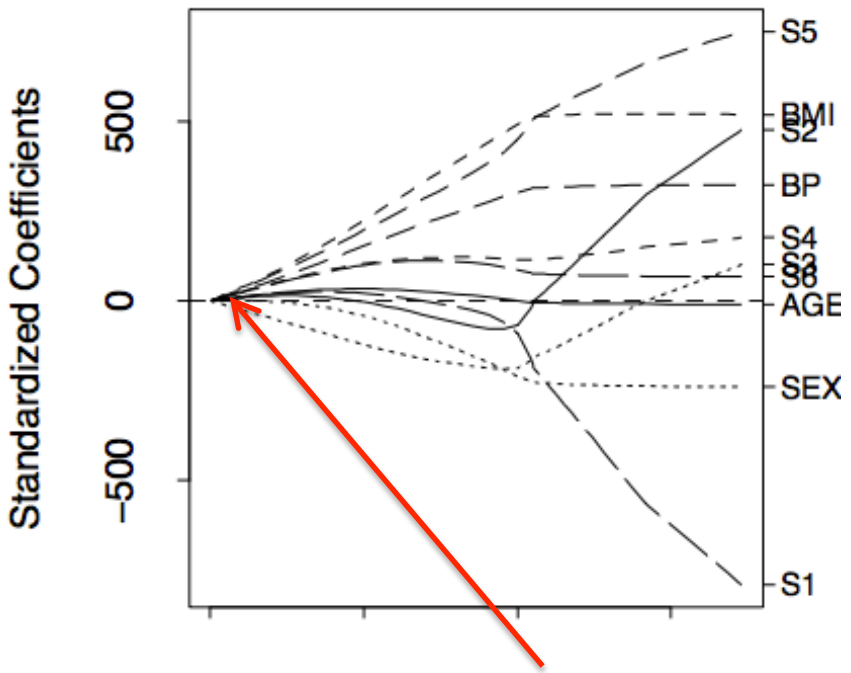
- Example – [E2004]

LARS



Coefficients become non-zero one at a time

Ridge regression



All coefficients become non-zero at the same time

LARS in DAPA

- Data collected by DAPA
  - System metrics as the predictor variables, e.g, CPU %, disk I/O
  - Application performance as the response variable, e.g., response time of a search request
- At each step, one metric is included into the learned model, i.e., its coefficient becomes non-zero
- Determine the stop condition such that an appropriate subset of metrics are included in the model (*Metric pre-selection*)

Metric pre-selection

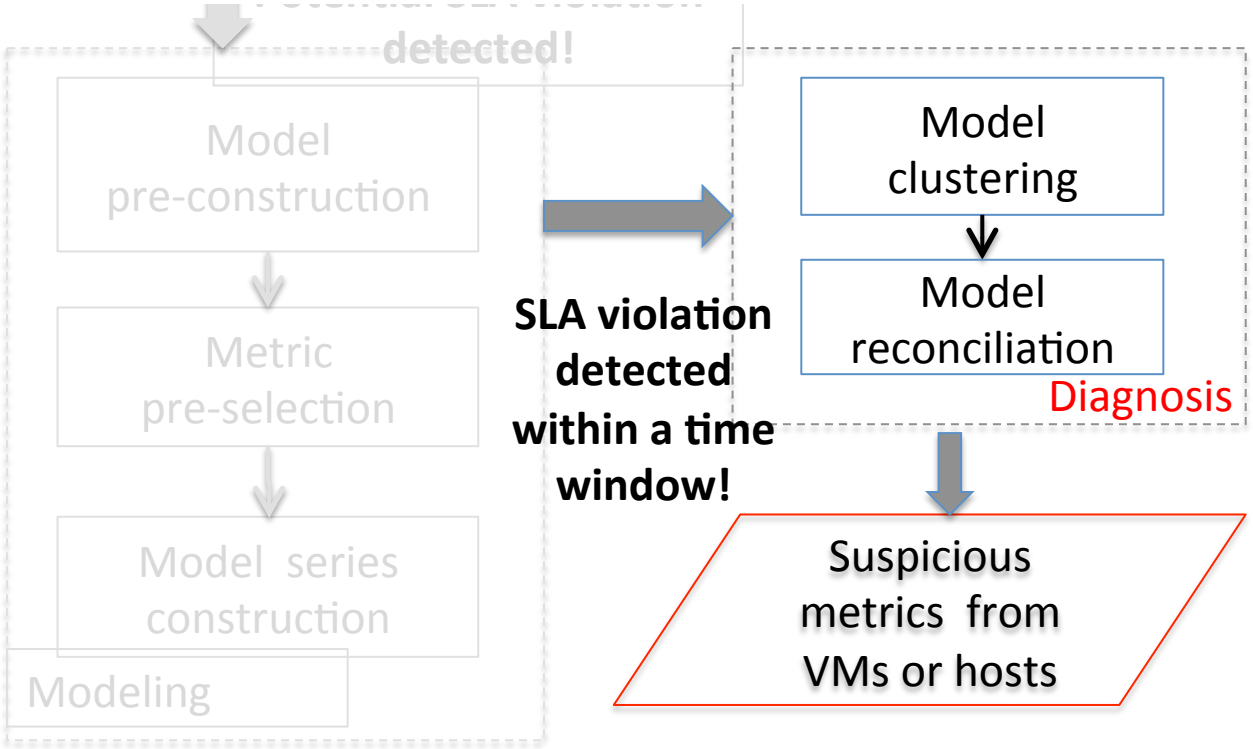
- A preliminary step before potential SLA violation
- $C_{p,i}$: the risk of including a metric in the model

```
1    Setup Regression stop condition;
     Setup corrThreshold;
     C_min = INT_MAX; inc = 0;
     While (inc not exceed stop condidtion)
5    Do
          C_p,i = including X_i in the set
          If C_p,i < C_min
               keep the metric in the output,
               improving model fit and reduce risk
10        Else
               X_i increase the risk
               inc++;
               Check the correlations between X_i
               and other metrics already in the set;
15             remove those exceeding the corrThreshold;
          C_min = C_p,i;
     End
```

Output the most important metrics for modeling the system

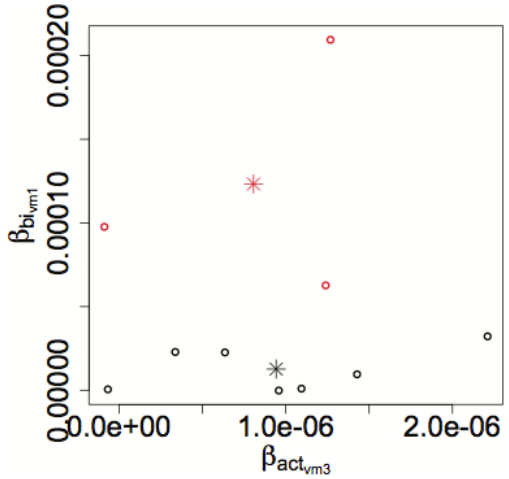
DAPA - Diagnosis

- Build a series of models using pre-selected metrics
- A true SLA violation triggers the diagnosis phase



Model clustering and reconciliation

- *k*-means clustering (k=2) classifies models into two clusters: SLA compliance and violation



Clustering models

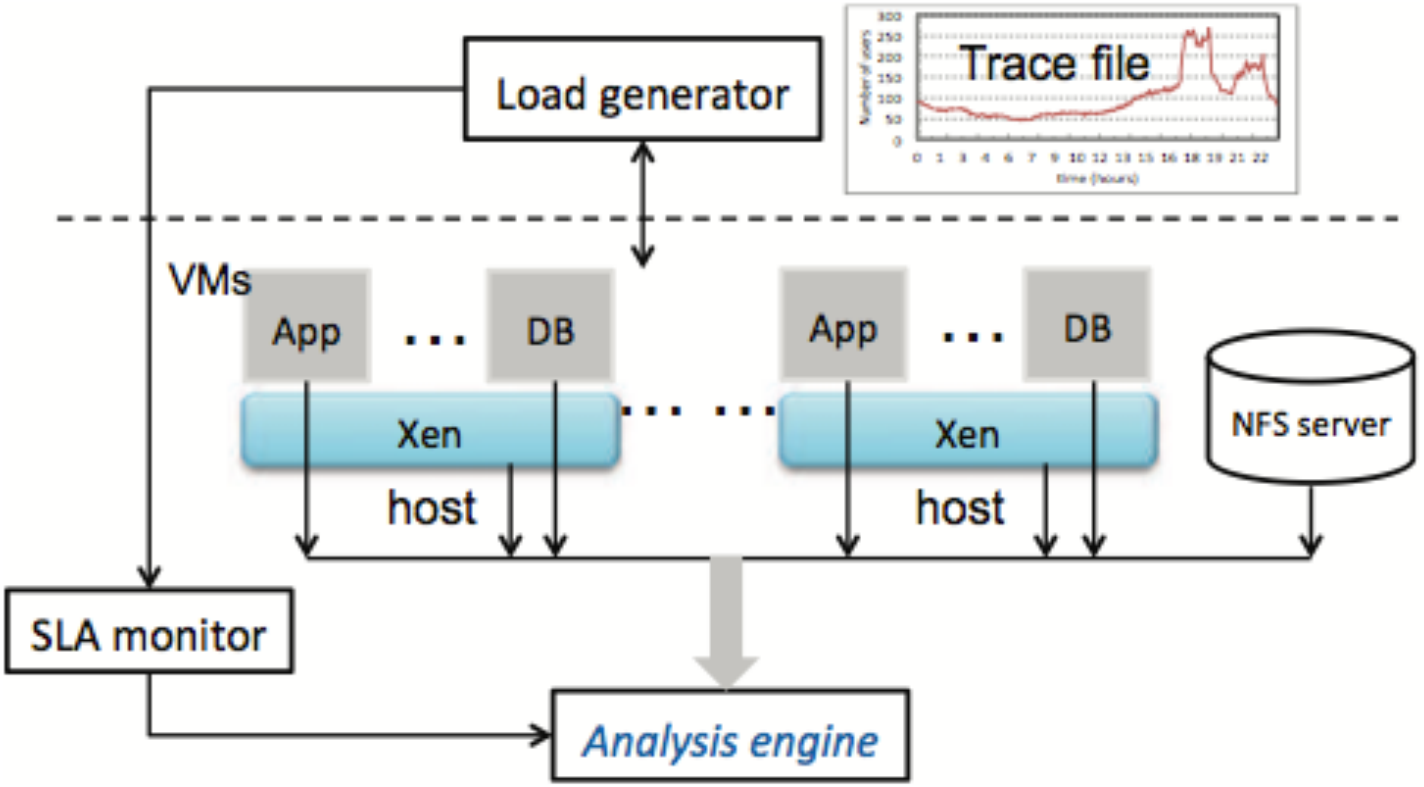
- Aggregate the data samples of the SLA violation cluster and create a new LARS model
  - Metrics ordered by their impacts to the SLA violation

Experiment setup

- Deploy DAPA prototype in the virtualized environment
- Inject several synthetic faults into the system, causing SLA violations
- Metric pre-selection is crucial
- Localize the resource bottleneck in a component related to the SLA violation

System architecture

- Xen 4.0 hypervisor and all VM Fedora 13
- Olio – a multi-tier social network application
- Trace driven workloads [A2000]



Data collection

- Application performance
- Critical system metrics
  - CPU, memory, disk, network, OS
- Runs in 5 second interval

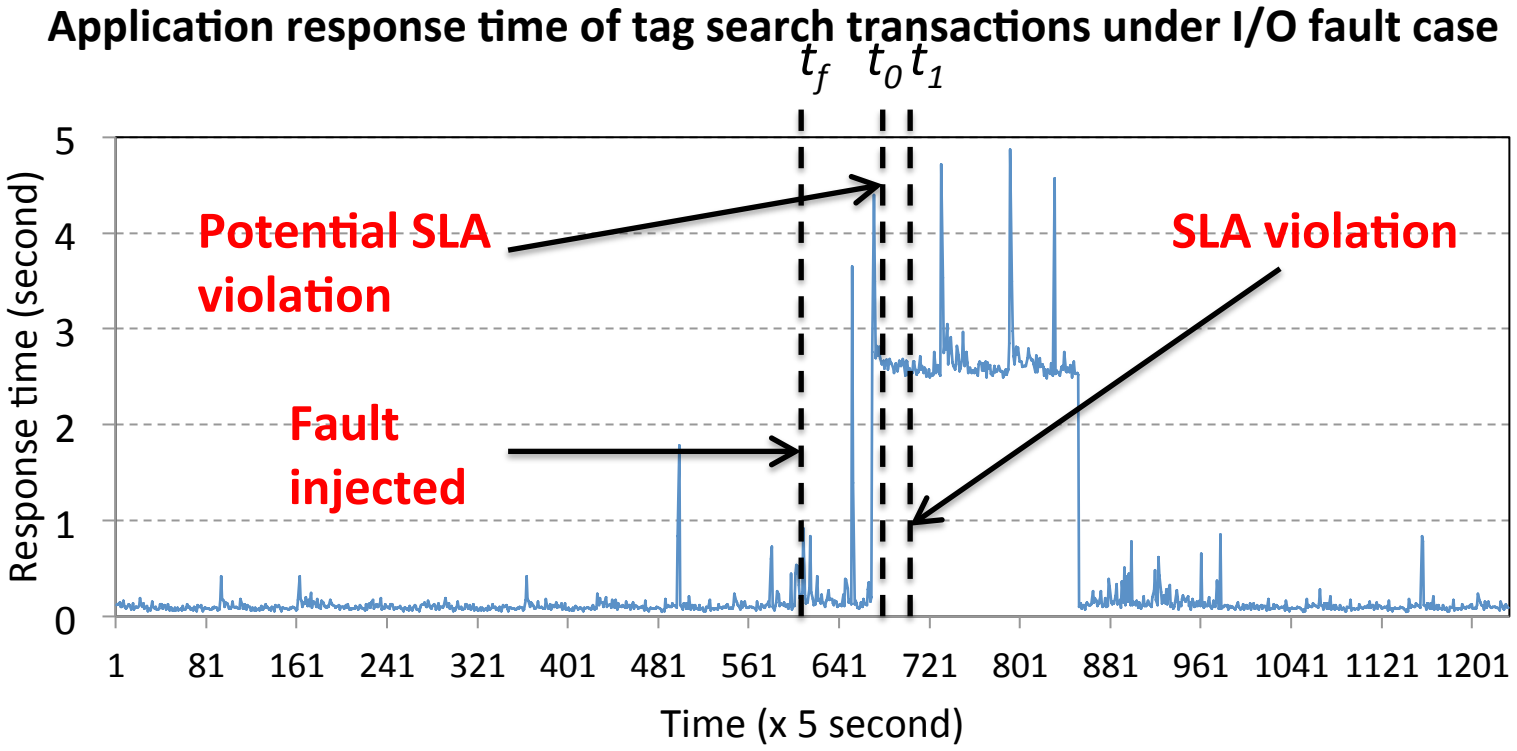
- **Application performance metrics is the tag search transaction response time**
- **SLA violation – " > 10% of the response time within one minute are greater than one second"**

Collected metrics

Category	Metrics	Description
CPU	uc	CPU utilization in user mode
	sysc	CPU utilization in kernel mode
	wait	CPU idle waiting for outstanding I/O requests
Memory	act	the amount of active memory
	inact	the amount of inactive memory
	si	amount of memory swapped in from disk
	so	amount of memory swapped out to disk
Disk	bo	# blocks sent to the block device
	bi	# blocks received from the block device
Network	rxp	transmitted packets
	txp	received packets
OS	cxt	# context switch
	in	# interrupts

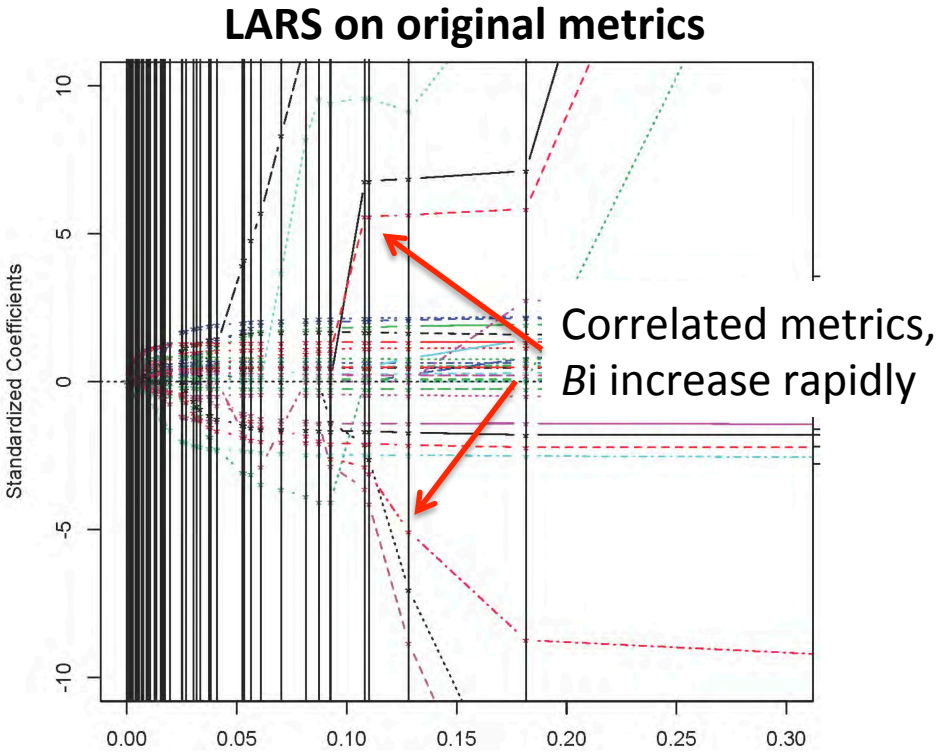
Synthetic faults injected

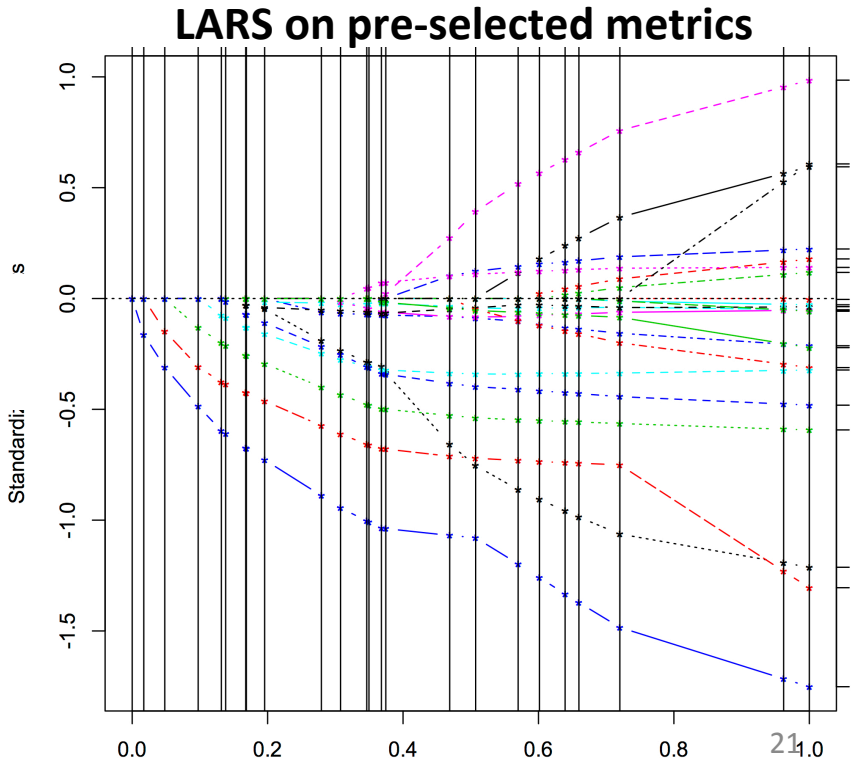
- Inject one single fault at a time
  - I/O intensive processes in an app-tier VM
  - Ballooning down the available memory of a DB VM
  - Add CPU-intensive VM to contend with an app-tier VM
- Each LARS model built on 120 data samples
- 10 models (100 minutes run) for each fault



Results – metrics pre-selection

- Reduce number of metrics by 1/3, from 54 to 32
- Eliminate highly correlated metrics
- The scales of Y-axis are different





Results – selected metrics

Injected faults	Top three metrics from DAPA	Description
Disk	io $wait_{vm1}$, bo_{vm1}, cxt_{vm3}	IO wait, block out, and cxt switch selected; intensive I/O activities on vm1 and vm3 affected
Memory	act_{vm4}, so_{vm4}, si_{vm4},	Active memory, swap out/in selected; vm4 short of memory
CPU	hc_{vm1}, hc_{vm2}, htx_{vm1},	Host CPU resource contention between vm1 and vm2;

Conclusion

- DAPA: a model-based performance diagnosis framework
  - Consider metric variations and scalability
  - Customized modeling techniques
  - Output ranked metrics related to performance loss
- Deploy with a multi-tier web application in a virtualized infrastructure
- Localize several bottlenecks causing the SLA violations

Future work

- Incorporate internal knowledge from various system layers of the virtualization platform
  - Gray-box approach
  - Differentiate root causes between misconfiguration of VM and interference from other VMs
- Global knowledge e.g., network topology
- Automatically finding the causal relationship between VM instances in the hosted application

Thank you! ☺

References

- [A2000] ARLITT, M., AND JIN, T. A workload characterization of study of the 1998 world cup web site. In *IEEE Networks* (May/June 2000), pp. 30–37.
- [B2003] BARHAM, P., ISAACS, R., MORTIER, R., AND NARAYANAN, D. Magpie: Online modeling and performance-aware systems. In *HotOS IX* (2003), pp. 85 – 90.
- [B2007] BAHL, P., CHANDRA, R., GREENBERG, A., KANDULA, S., MALTZ, D. A., AND ZHANG, M. Towards highly reliable enterprise network services via inference of multi-level dependencies. In *Sigcomm* (2007), pp. 13–24.
- [C2004] COHEN, I., CHASE, J. S., GOLDSZMIDT, M., KELLY, T., AND SYMONS, J. Correlating instrumentation data to system states: a building block for automated diagnosis and control. In *OSDI* (2004), pp. 231–244.
- [E2004] EFRON, B., HASTIE, T., JOHNSTONE, I., AND TIBSHIRANI, R. Least angle regression. *The Annals of Statistics 32* (2004), 407–499.
- [K2005] KOMPELLA, R. R., YATES, J., GREENBERG, A., AND SNOEREN, A. C. Ip fault localization via risk modeling. In *NSDI* (2005), pp. 387–390.
- [P2009] PADALA, P., HOU, K.-Y., SHIN, K., ZHU, X., AND UYSAL, M. Auto- mated control of multiple virtualized resources. In *Eurosys* (2009), pp. 13– 26.

Backup

Results – metrics ranking (I/O fault case)

