Automatic Stress Testing of Multi-tier Systems by Dynamic Bottleneck Switch Generation

G. Casale[1], A. Kalbasi[2], D. Krishnamurthy[2], J. Rolia[3]

[1] SAP Research, UK
[2] University of Calgary, Canada
[3] HP Labs Bristol, UK

Presenter: Giuliano Casale

Middleware,
Urbana, IL, Dec 4th, 2009

Motivation

- Stress testing of enterprise multi-tier systems
  - Difficult to validate systems before deployment
    - Benchmarks with repetitive behavior too simplistic
  - Effects of real workloads hard to characterize and recreate
    - SAP Business suite application: >57,000 transactions types

- Contribution
  - Methodology to define more realistic stress workloads

- Requirements
  - Custom workload mixes
  - Realistic time-varying resource usage patterns
  - Automatic benchmark generation

Problem Statement

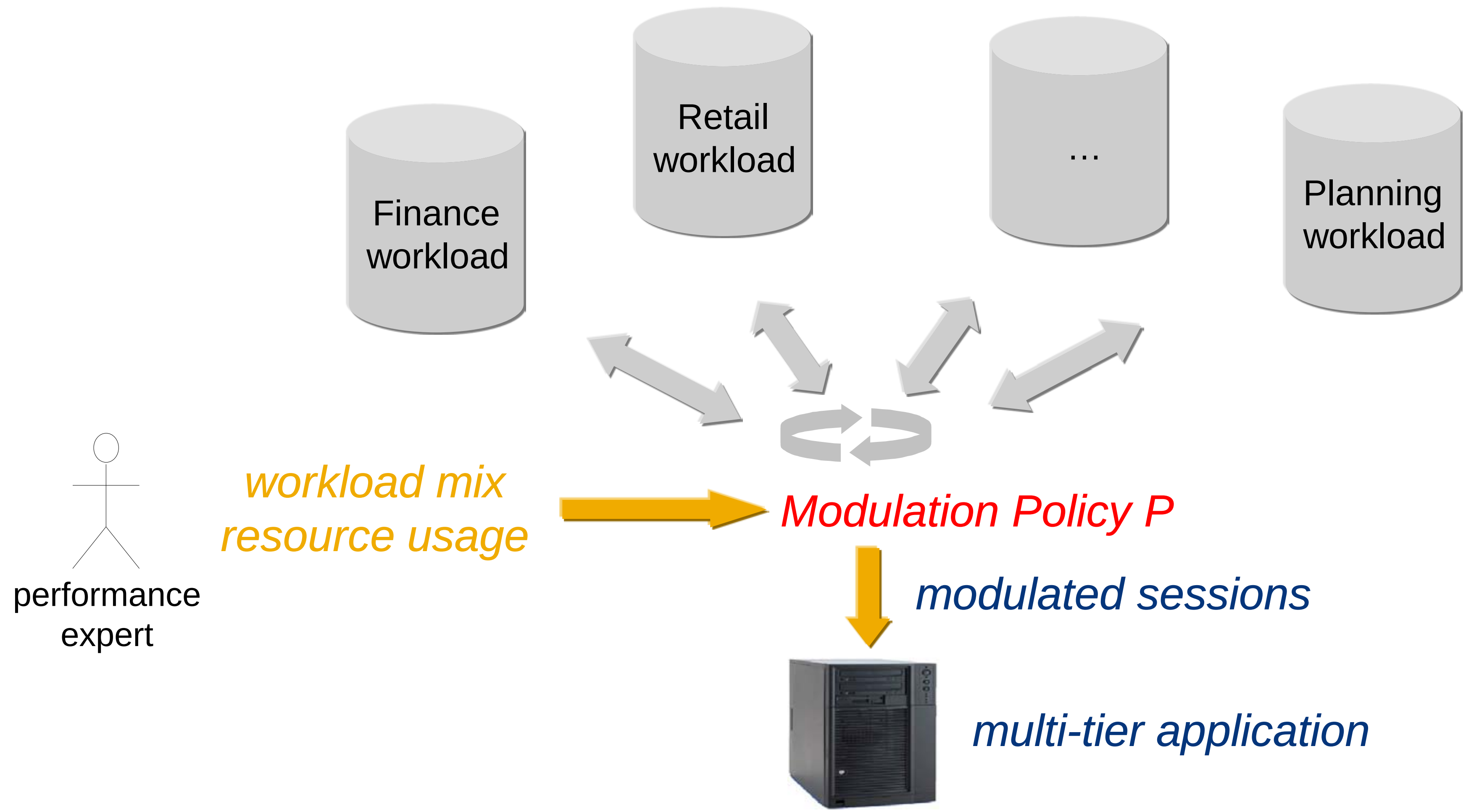
- Issues in real-world stress testing
  - Load often submitted by commercial or ad-hoc tools
  - Often impossible or difficult to extend these tools directly
  - A number of workloads already in place for stress testing

- Idea
  - Inter-leave execution of pre-existing workloads/benchmark

- Problem
  - How to achieve desired resource usage patterns?

Stress testing system

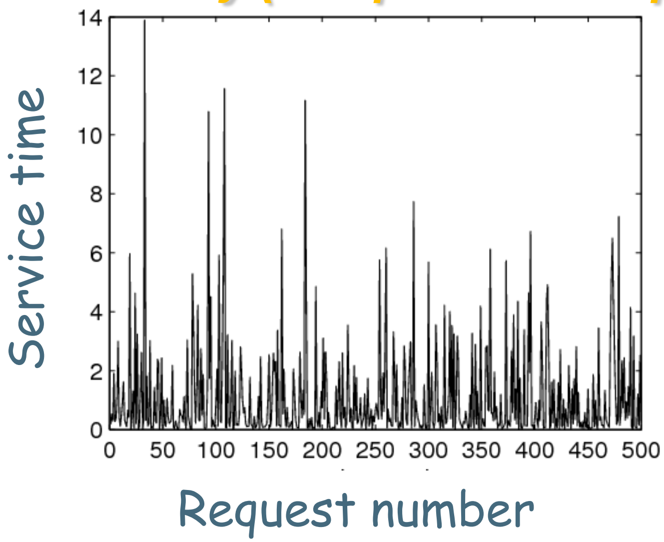
- Each pre-existing benchmark provides workload



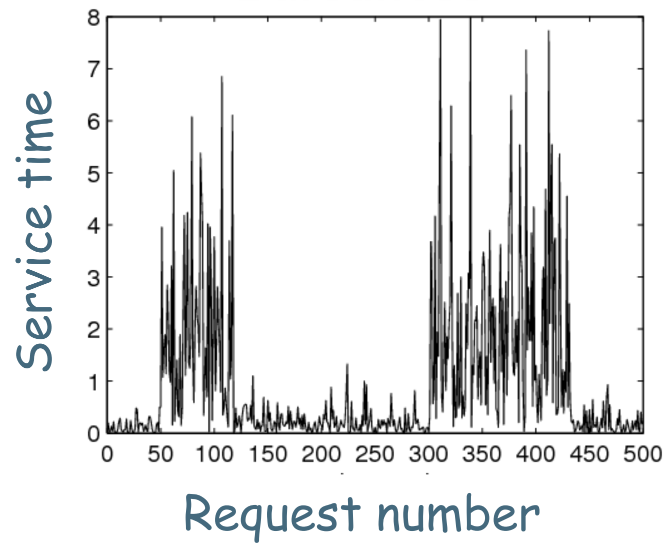
Time-Varying Resource Usage

- Burstiness/variability critical factor for scalability
- Service time burstiness frequent in multi-tier systems
  - Smirni, Cherkasova, et al., *Performance’07, Middleware’08*
  - Root causes: multiclass workloads, caching, DB locking, …

Variability (Independent Reqs)



Burstiness



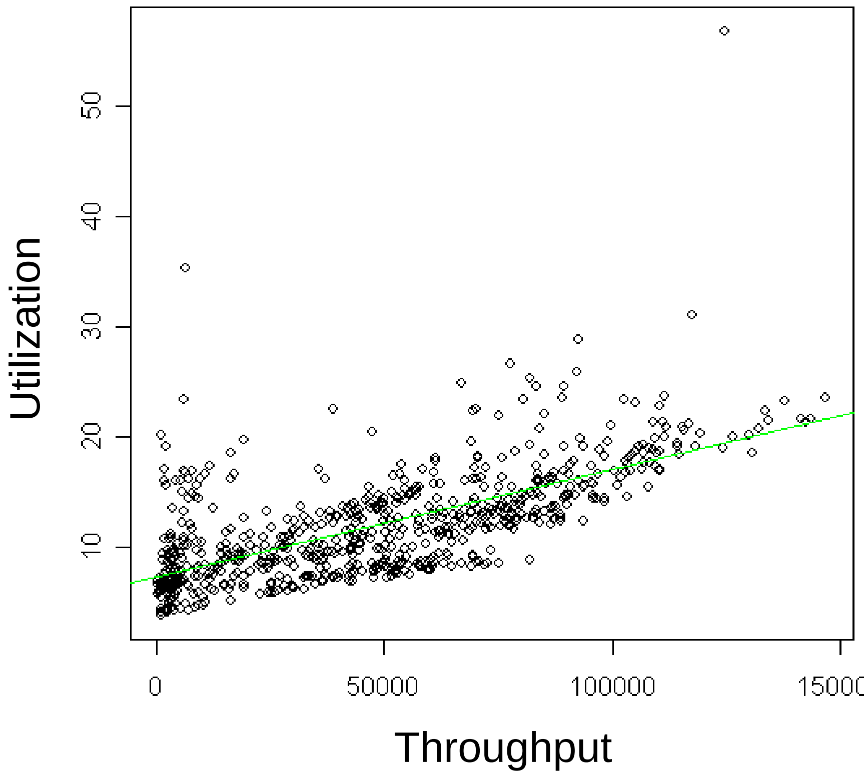
- Burstiness can lead to large performance degradation
  - Effect produced by bottleneck switch between resources

Methodology

- Demand characterization and estimation
  - Model service demand placed by requests on each resource
  - Compact description based on Markov models
- Composition step
  - Predict impact of modulation of pre-existing benchmarks
- Benchmark generation
  - Automatic generation of target workload
  - Non-linear programming approach

Service Demand Estimation

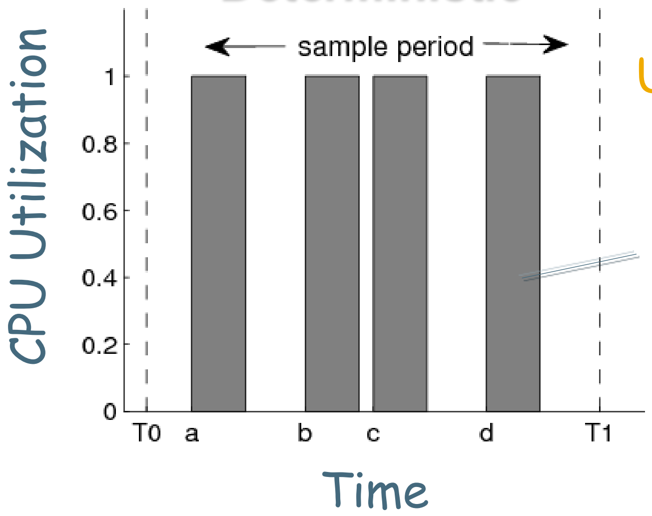
- Set of tput and utilization samples for each pre-ex. wrkload
- Mean: linear regression of collected samples
  - first application in 1970s
  - applicable also to systems with multiple transaction types



Service Demand Estimation

- Variance: needed for controlling burstiness/variability
  - Regression often does not apply to variance estimation
  - Main problem: information loss due to sampling

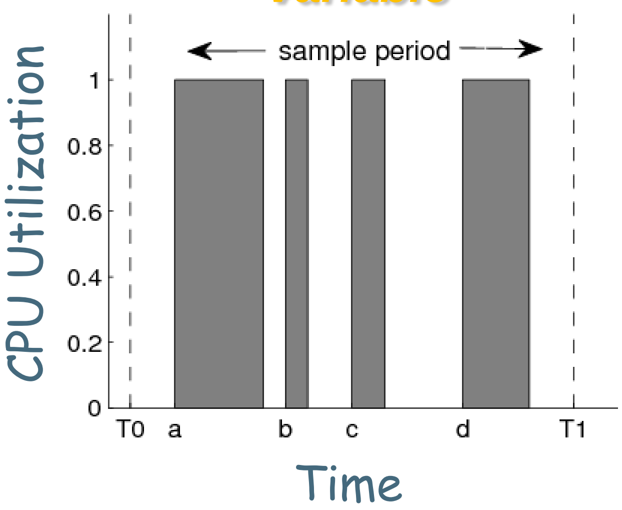
Deterministic



Identical Utilization Samples

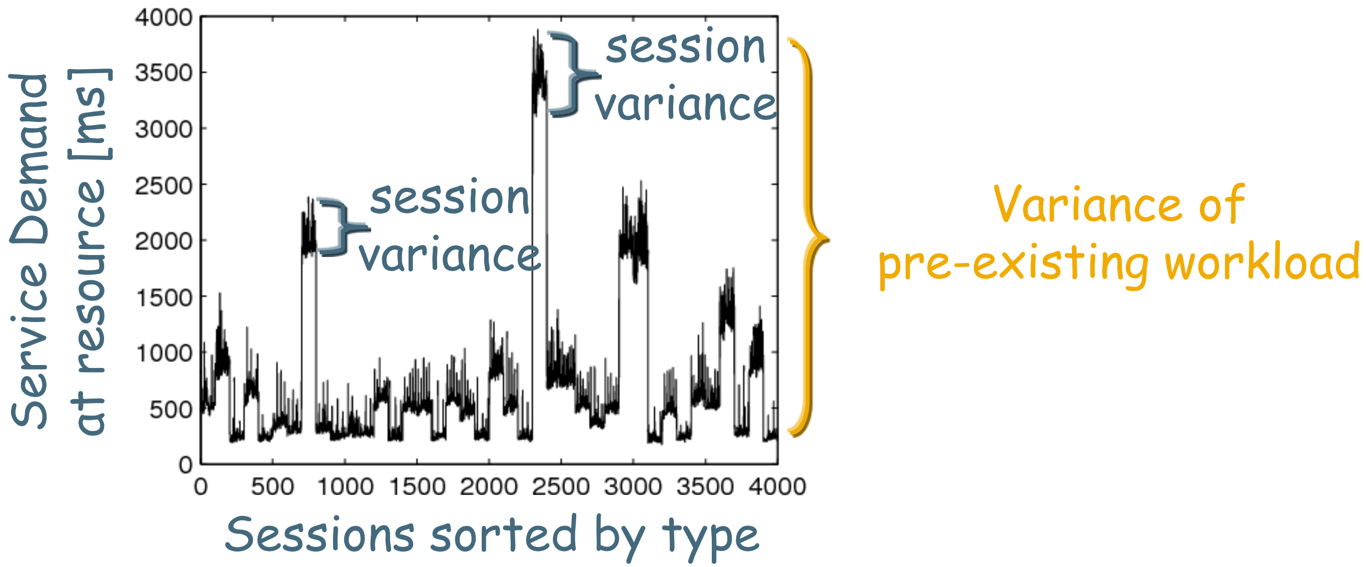
Busy Time

Variable



Variance Characterization

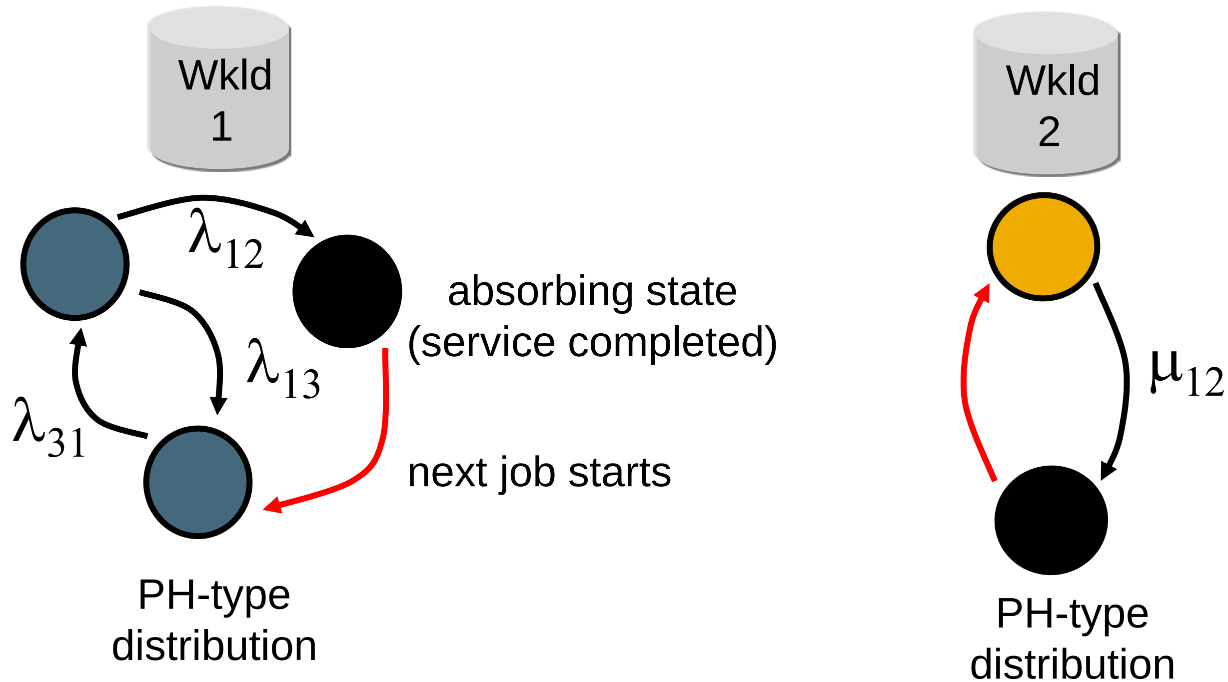
- Approximation
  - Each group has a set of pre-existing session types
  - Sessions within a group are drawn randomly



- Approximation: group variance ≈ variance of session type mean
- Similar approximation holds for skewness, kurtosis, …

Service Demand Modeling

- Demand distribution described by a phase-type model
  - a class of continuous-time Markov chains
  - time to complete service = time to reach absorbing state
- State jump rates fitted from demand moments



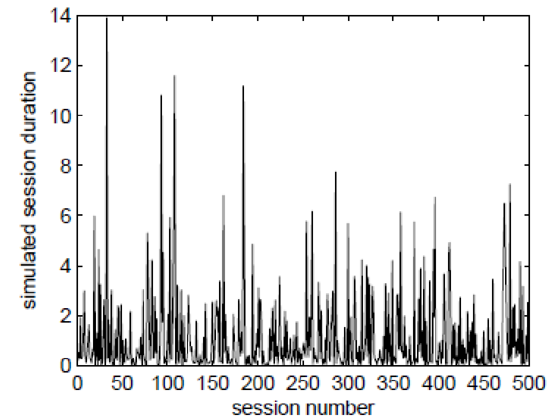
Session Submission Policy

- Policy defined as discrete-time Markov chain
  - Workload mix: $\gamma = \gamma \mathbf{P},$
  - Assume G pre-existing workloads

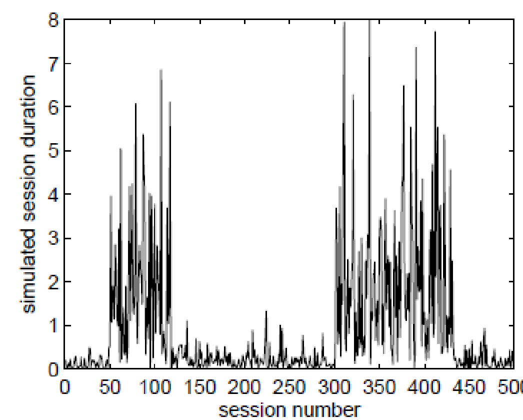
$$\mathbf{P}^{trad} = \begin{bmatrix} 0.50 & 0.50 \\ 0.50 & 0.50 \end{bmatrix}$$

$$\mathbf{P} = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,G} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,G} \\ \vdots & \vdots & \ddots & \vdots \\ p_{G,1} & p_{G,2} & \cdots & p_{G,G} \end{bmatrix}$$

$$\mathbf{P}^{burst} = \begin{bmatrix} 0.99 & 0.01 \\ 0.01 & 0.99 \end{bmatrix}$$



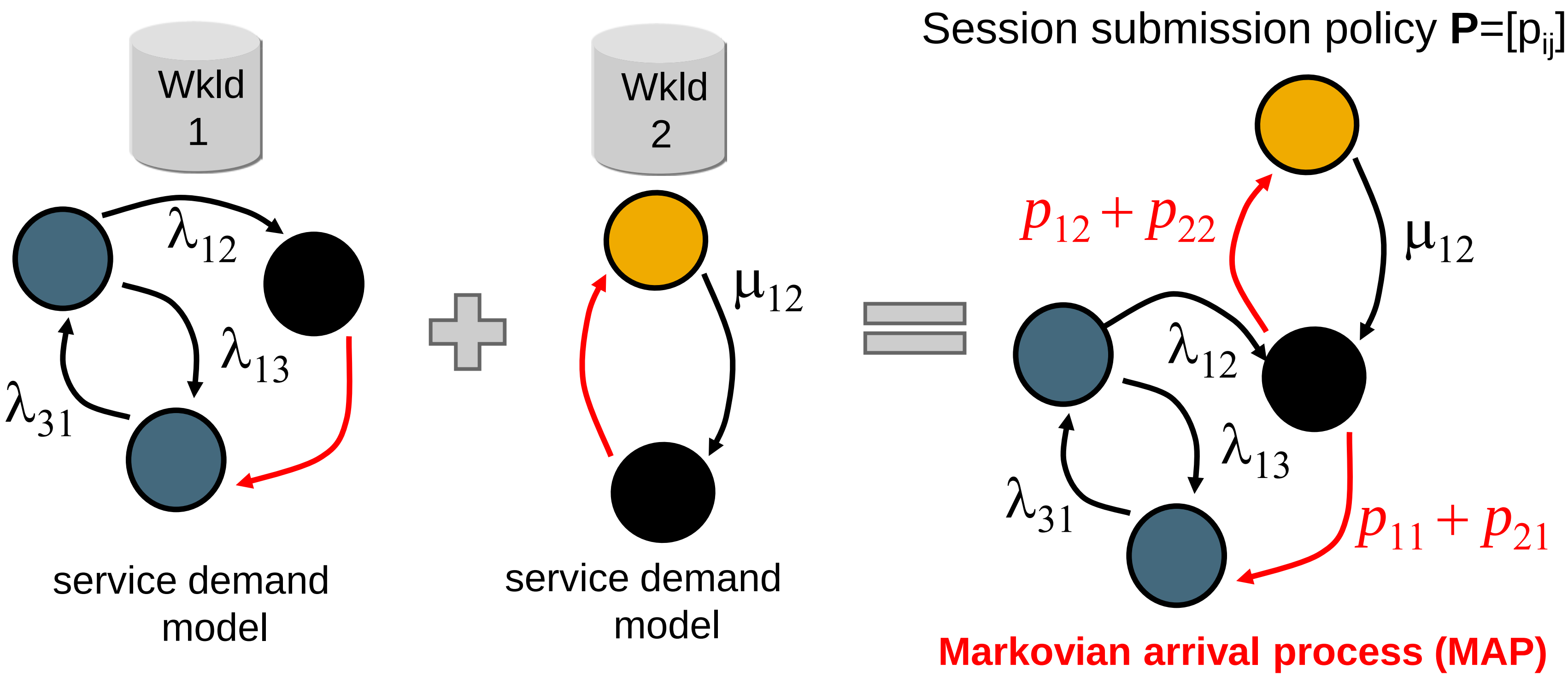
(b) Traditional policy



(a) Bursty policy

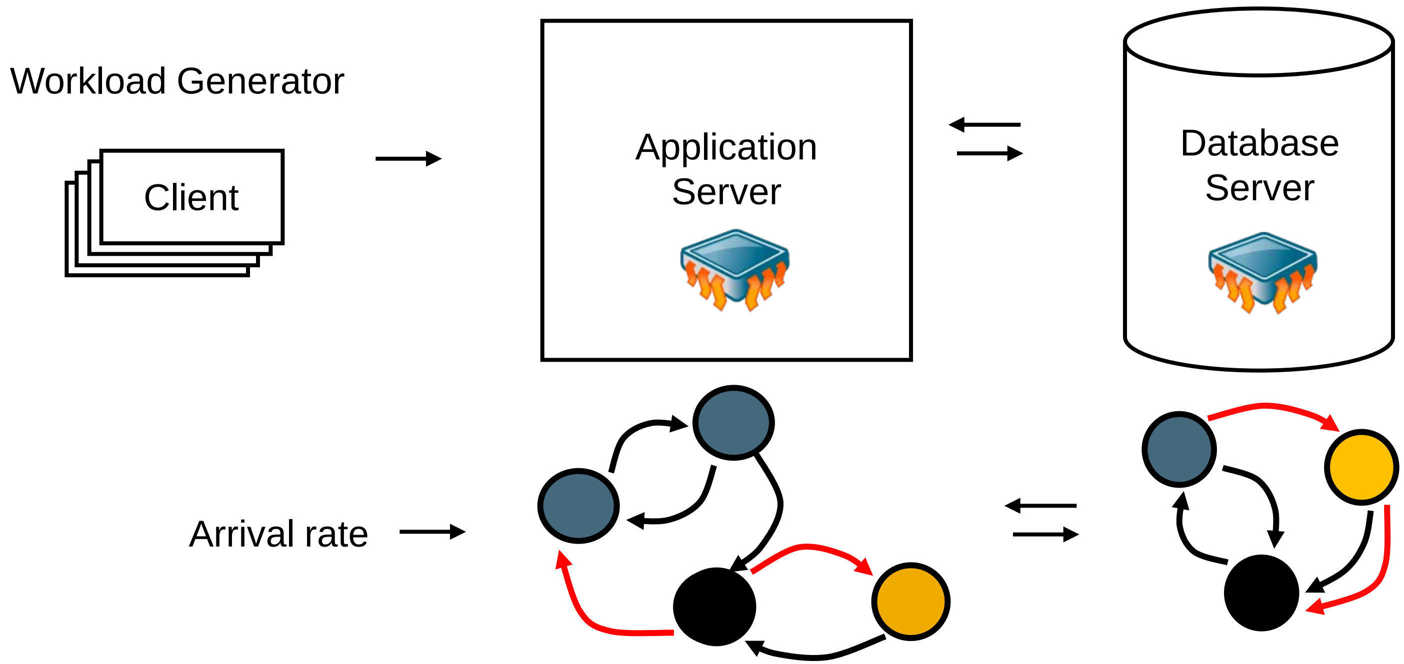
Session Submission Policy

- Policy defined as discrete-time Markov chain
  - compositional properties of Markov models



Resource Usage Prediction

- Example: three-tier application

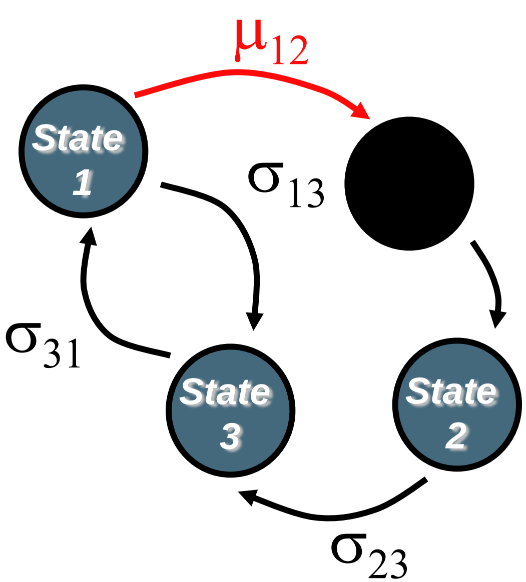


- MAPs building block for resource usage prediction
  - Distribution/burstiness of aggregate service demand
  - Distribution/burstiness of utilization (controls bottleneck switch)
  - Distribution/burstiness of response times

Markovian Arrival Process

- Acts as predictor of demand for modulated workload
  - distribution of service times given moments
  - estimate persistence of bursts at resource

- Example of analytical description (absorbing states hidden)



$$D_0 = \begin{pmatrix} -\Sigma\ldots & 0 & \sigma_{13} \\ 0 & -\Sigma\ldots & \sigma_{23} \\ \sigma_{31} & 0 & -\Sigma\ldots \end{pmatrix} \qquad D_1 = \begin{pmatrix} 0 & \mu_{12} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Invisible Transitions ***Service completion***

$$\Pr[S=t] = 1 - \gamma D_1 e^{D_0 t}\mathbf{1} / E[S],$$

Policy Generation

- User Input
  - Workload mix (% session types)
  - Resource usage pattern
    - For example, increase burstiness level **I** in DB utilization
- Policy generation
  - Search for modulation policy **P** = [p_{ij}] achieving targets
  - Nonlinear optimization program

$$\underbrace{\min_{\mathrm{P}} (\mathrm{I} - \mathrm{I}_{target})^2}_{\text{Target burstiness}}$$

$$s.t. \quad \underbrace{\begin{aligned} \gamma &= \gamma \mathbf{P}, \\ \Pr[S = t] &= 1 - \boldsymbol{\gamma} D_1 e^{D_0 t} \mathbf{1} / E[S], \\ &\ldots \end{aligned}}_{\text{Modulation effects characterization}}$$

Example: TPC-W system

- Benchmarking solution for multi-tier systems
  - standard three tier architecture (clients, front server, DB)
  - 3 existing workloads: browsing, ordering, shopping

- Experimental Validation
  - Open workload mostly of ordering and shopping (>95%)
  - Ordering and shopping do not have burstiness
  - Is a bursty combination of these workloads harmful?

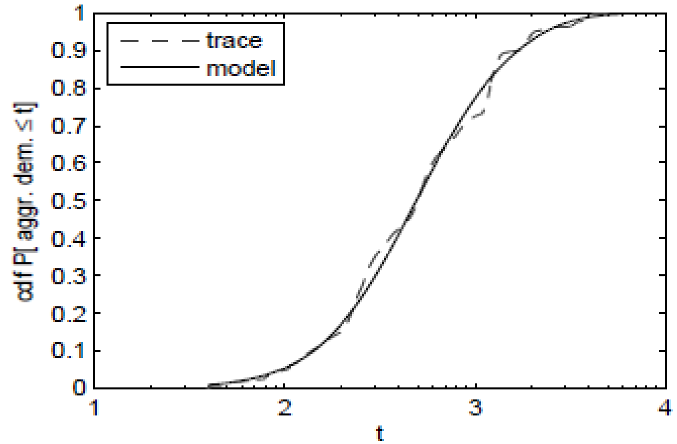
- Model of service demand is 15-state MAP
  - Policy **P** computed in less than one minute

Quality of Demand Models

- Automatic fitting of PH-type model for existing workloads

	front server demand			*DB server demand*		
shopping	*mean*	*CV*	*skew*	*mean*	*CV*	*skew*
measured	0.290	0.575	2.671	0.097	7.590	4.509
PH-type	0.290	0.575	1.665	0.097	7.590	4.509
ordering	*mean*	*CV*	*skew*	*mean*	*CV*	*skew*
measured	0.131	0.805	1.797	0.623	1.761	2.530
PH-type	0.131	0.806	1.798	0.623	1.761	2.531

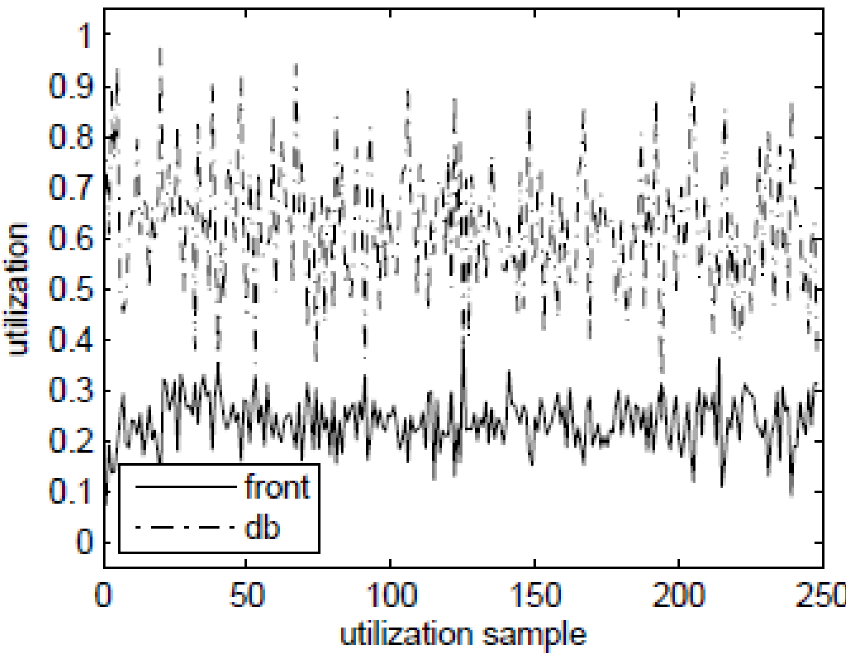
  - good matching of distribution/burstiness of modulated wkld



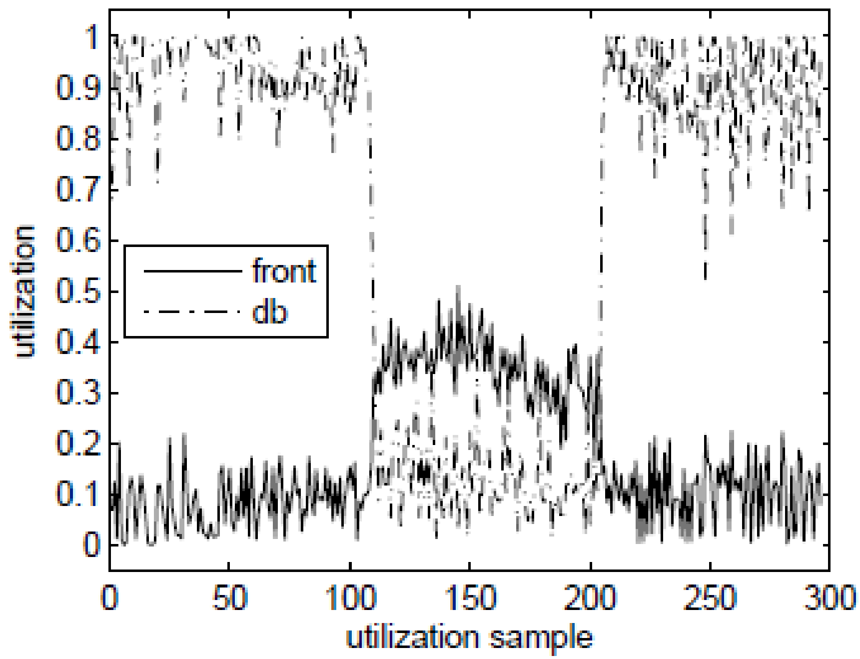
(a) CDF Aggregate Demand (sec)

Burstiness Effects on Utilization

- High burstiness results in bottleneck switch effects
  - deleterious effects on performance due to transient



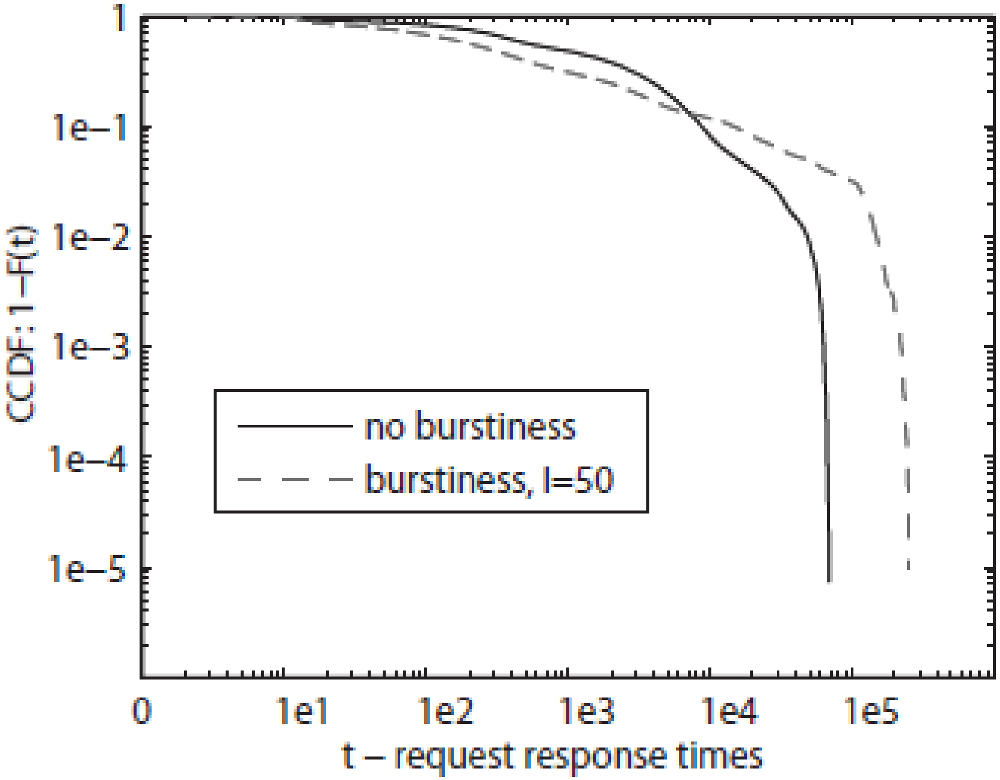
(a) $I = 1.14$, No Burstiness



(b) $I = 50$, High Burstiness

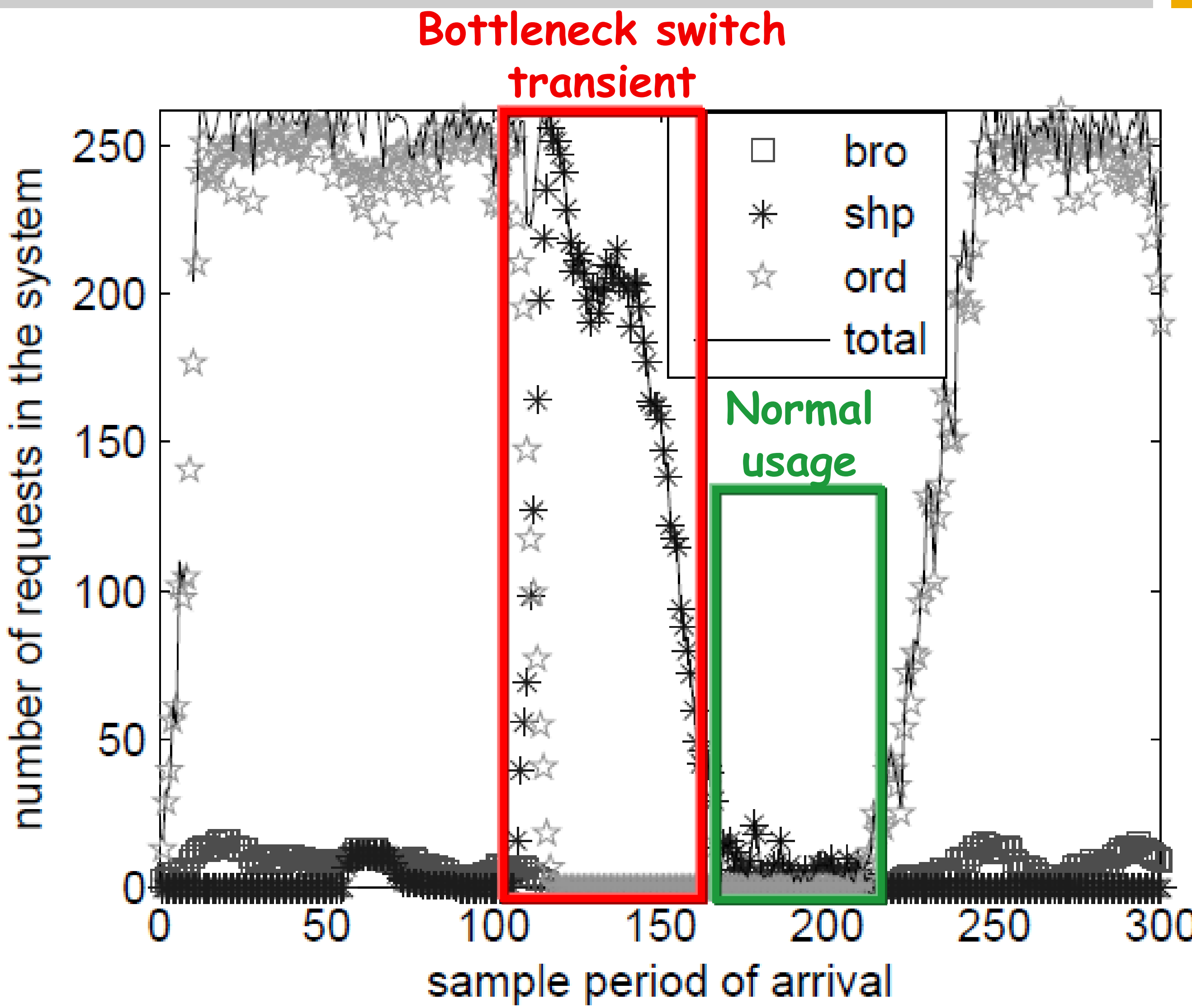
Burstiness Effects on Response

- Comparison of response times with and without burstiness
  - No-burstiness indications would lead to under-sizing



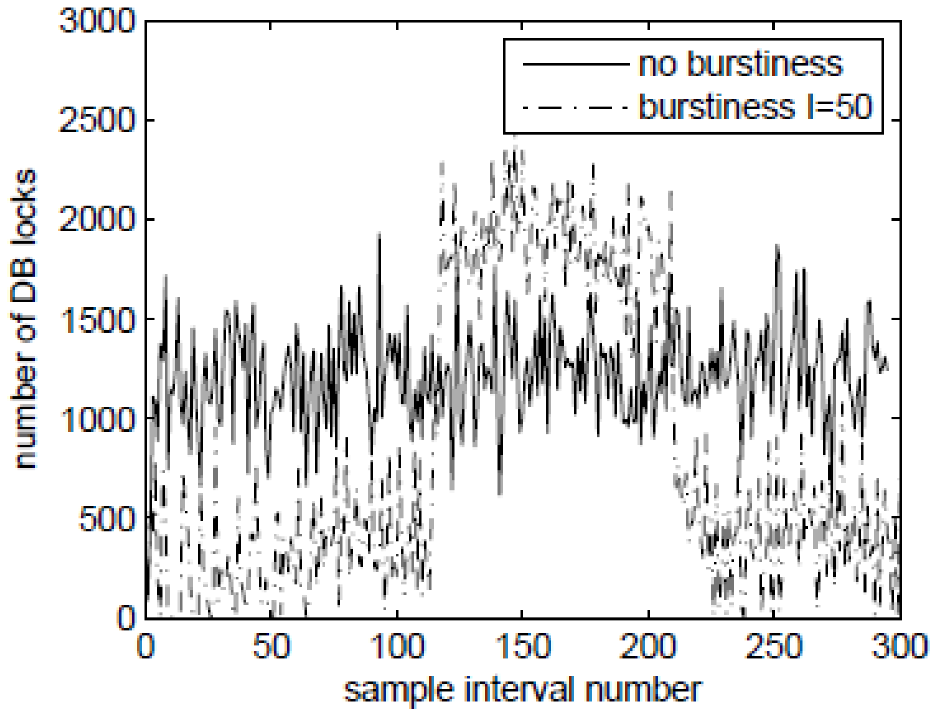
(c) CCDF Response Times

Observation on Real Systems



Effects on DB Contention

- Database contention stresses by bottleneck switch
  - System stressed under more comprehensive set of cases



(d) Locks

Conclusion and Future Work

- Realistic stress-testing of multi-tier systems
  - Reuse of set of pre-existing workloads
  - Custom resource usage patterns

- Future work
  - Validation on industrial systems (SAP Business Suite)
  - Interaction with admission control systems
  - Evaluation of effectiveness on other resources (e.g., disk)

Thanks!