

# Automated Parameterization of Performance Models from Measurements

**Giuliano Casale**

*Imperial College London, UK*

**Simon Spinner**

*University of Würzburg, Germany*

**Weikun Wang**

*Imperial College London, UK*

*Tutorial @ ICPE 2016, Delft, the Netherlands, March 13, 2016*



Common parameters in performance models:

- **Service demand** of a request
  - CPU time, bandwidth consumed, ...
- **Arrival rate** of requests
- Applications
  - Automated performance modelling
  - Resource cost splitting
  - Performance anomaly detection
  - ...

# Example: Queueing Modelling

JMVA - Product form queueing network solver - 3c\_3stat\_mixed.jmva

File Action Help

Algorithm: MVA

Classes Stations Service Demands Reference Station What-if Comment

**Demands**

**Service Demands**  
Input service demands of each station and class.  
If the station is "Load Dependent" you can set the service demands for each number of customers by double-click on "LD Settings..." button.  
Press "Service Times and Visits" button to enter service times and visits instead of service demands.

*	Class1	Class2	Class3
Station1	5.000000	0.000000	0.000000
Station2	1.000000	1.100000	1.100000
Station3	1.600000	2.100000	0.500000

it, right-click to save it

Performance index: Residence Times

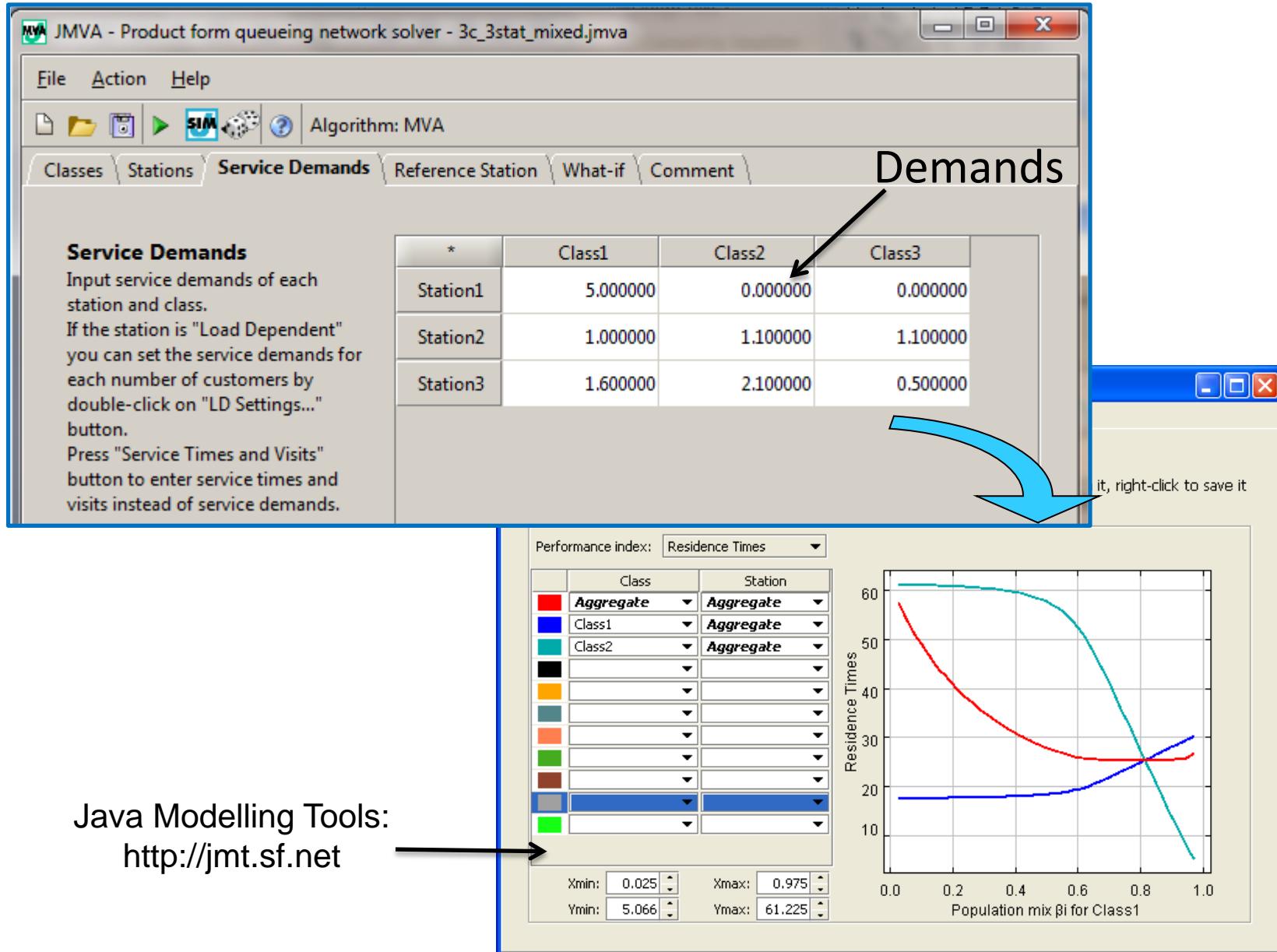
	Class	Station
Aggregate	Aggregate	Aggregate
Class1	Aggregate	Aggregate
Class2	Aggregate	Aggregate
	▼	▼
	▼	▼
	▼	▼
	▼	▼
	▼	▼
	▼	▼
	▼	▼
	▼	▼
	▼	▼
	▼	▼

Residence Times

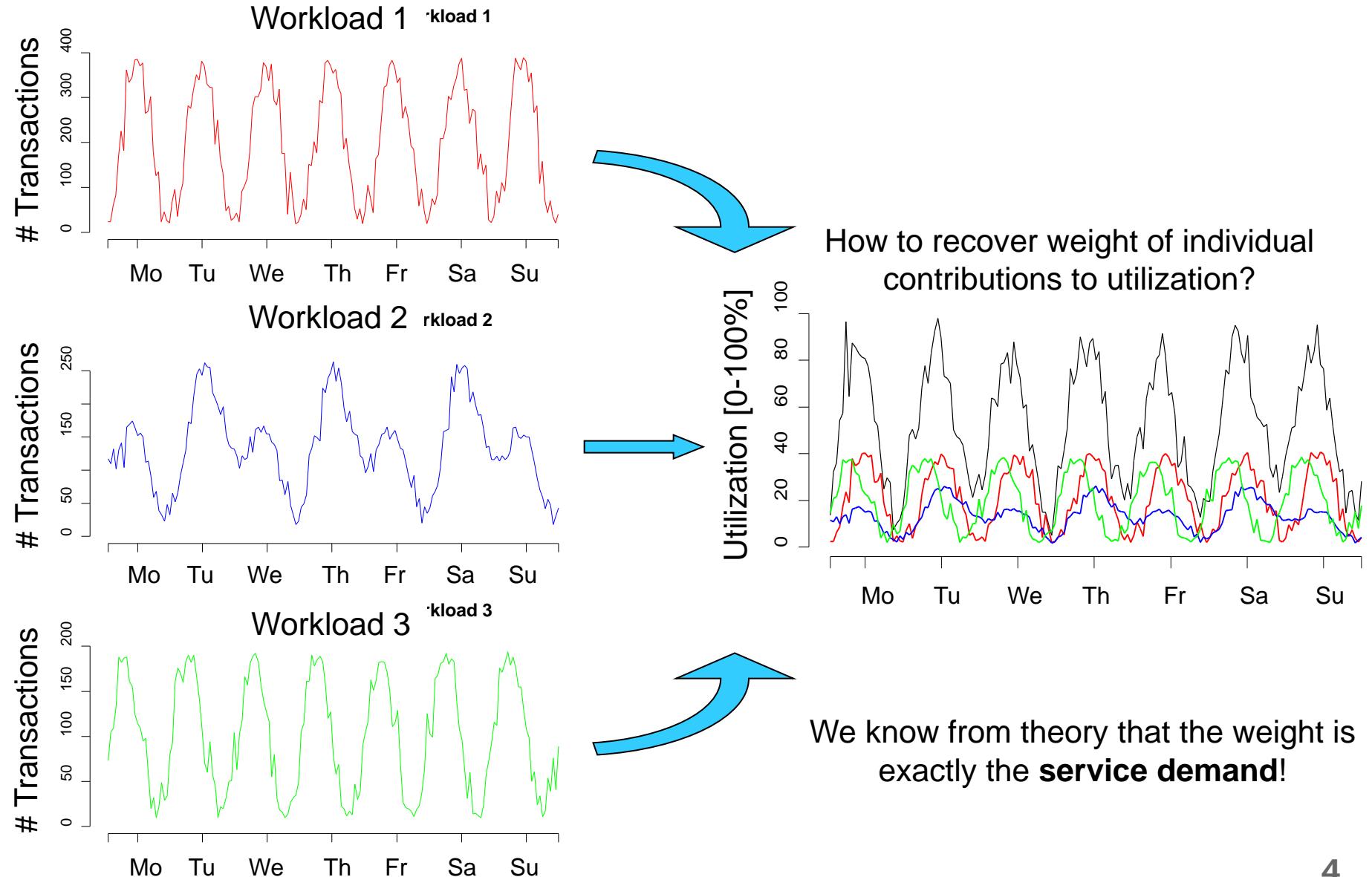
Xmin: 0.025 Xmax: 0.975  
Ymin: 5.066 Ymax: 61.225

Population mix  $\beta_i$  for Class1

Java Modelling Tools:  
<http://jmt.sf.net>

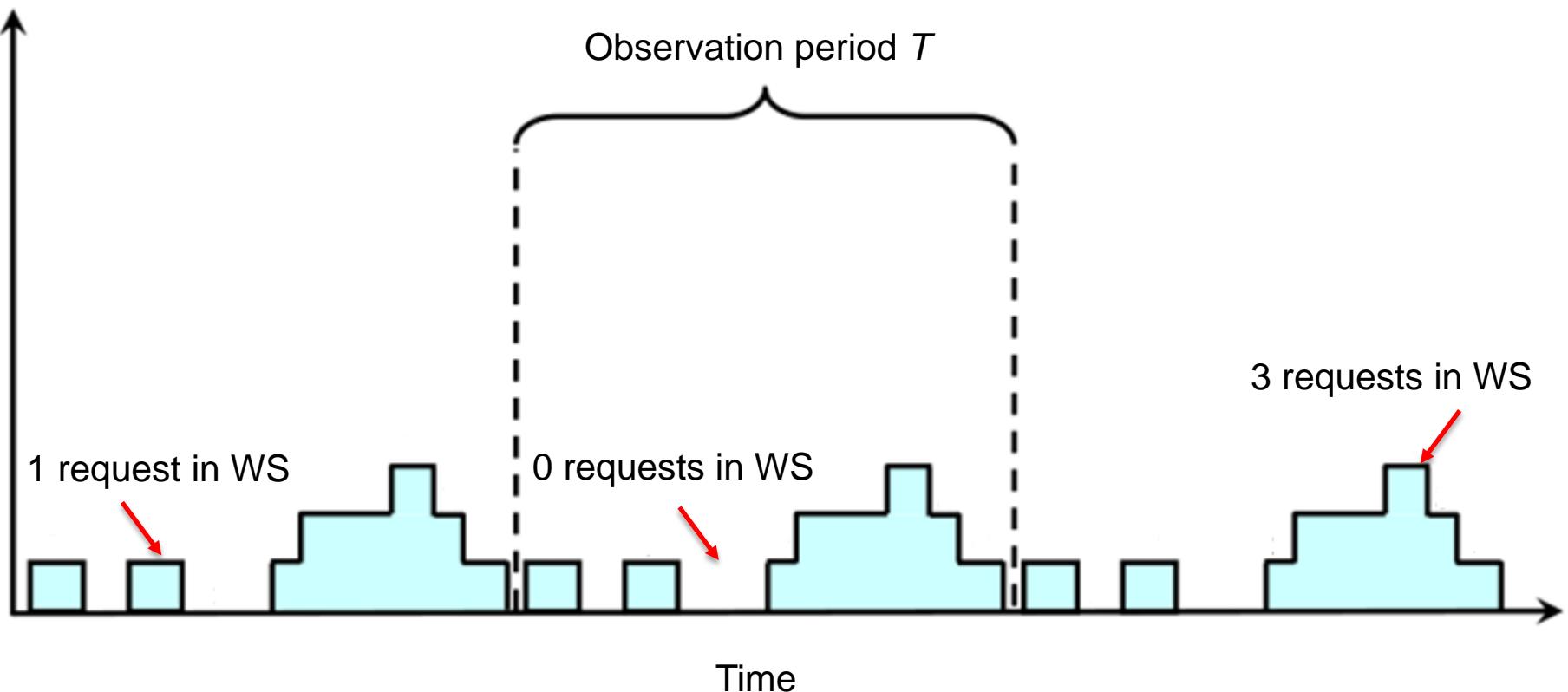


# Example: Cost Splitting



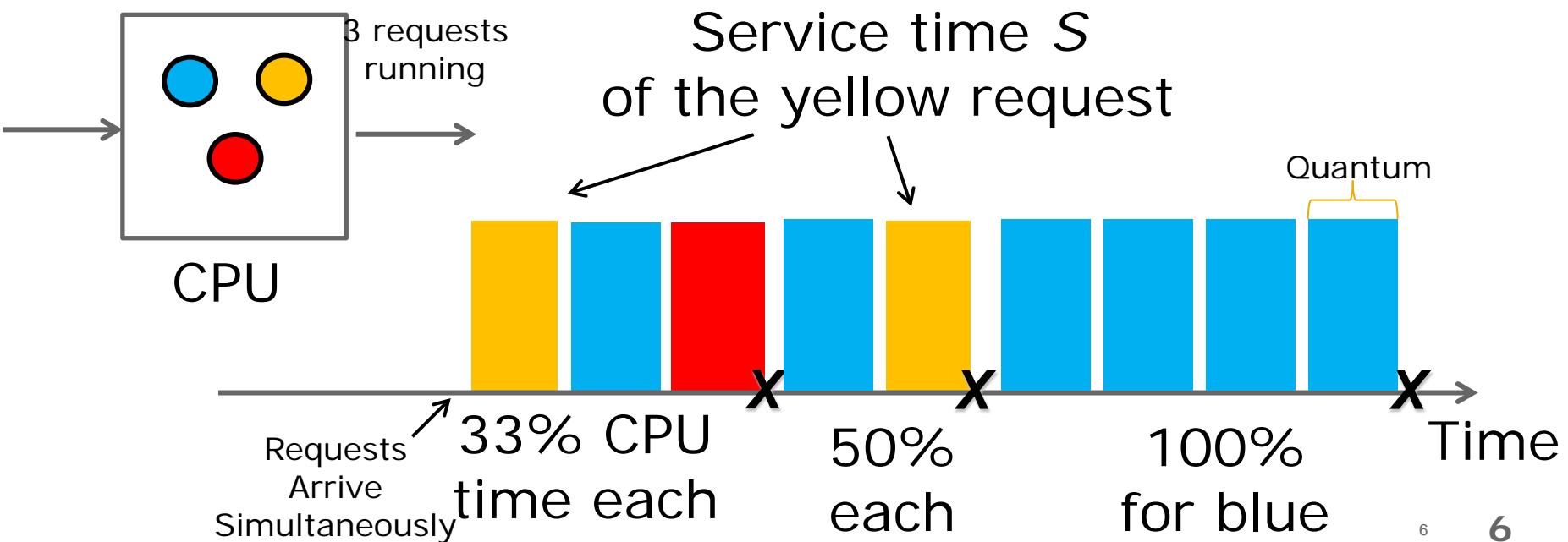
# A Typical Challenge

HTTP Requests  
in the WS  
(Web Server)



# A Typical Challenge

- ❑ OS schedules jobs in round robin
  - If  $n$  requests run simultaneously, each will approximately receive  $1/n$  of the CPU time
  - Process Sharing is a round robin where the quantum of time assigned to each request is infinitesimal



# Tutorial Agenda

- Introduction
- Demand Estimation
  - Utilization-based
    - LibReDE tool
  - Response-based
  - Queue Length-based
    - FG tool
- Comparison Study & Case Studies
- Arrival Rate Estimation
  - M3A tool

# Demand Estimation

# Overview

- Estimation Approaches (first part)
  - Simple
  - Utilization
  - Response Time
- LibReDE demo
- Estimation Approaches (second part)
  - Queue Length

- Trivial approximation:  $D_{i,c} \approx R_c$
- Assumptions
  - resource dominates system response time
  - waiting time in queue  $\ll D_{i,c}$
- Only applicable at low resource utilization

- Basic operational law:

$$D_{i,c} = \frac{U_{i,c}}{X_{0,c}}$$

- Partial utilization  $U_{i,c}$  is hard to derive

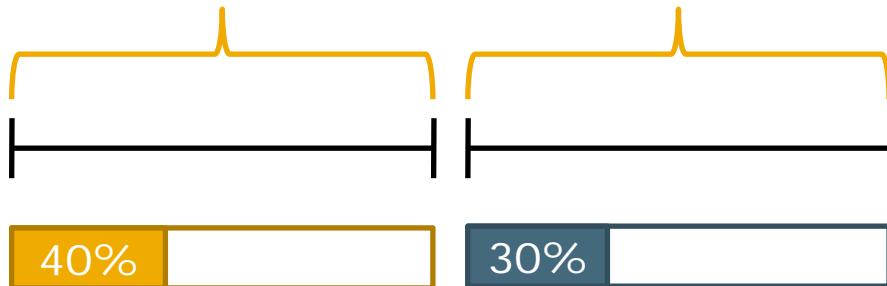
- Operating system: per-process statistics
  - Profilers: high-overheads

- 2 alternative solutions:

- Controlled experiment
  - Partitioning

# Controlled Experiment

- Measurement Interval
- CPU Utilization



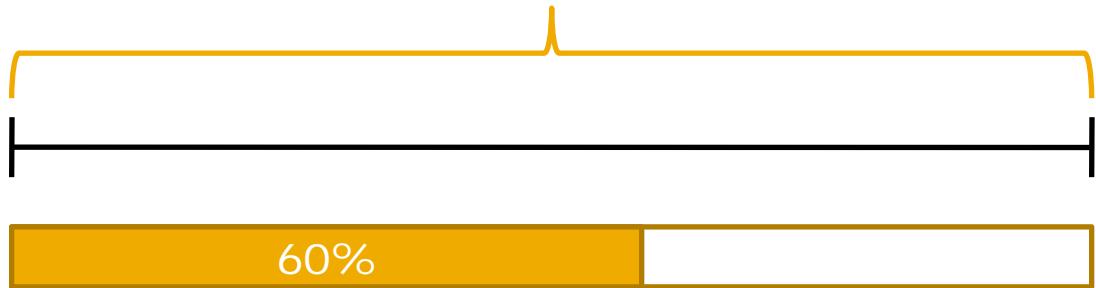
- Requests executed in separate experiments
- Resource Demand
- Problems:



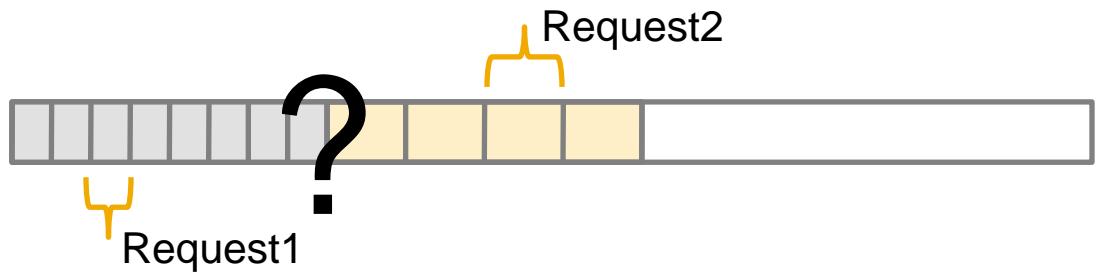
- Not applicable at runtime
- Mutual interference

# Partitioning

- Measurement Interval
- CPU Utilization



- Mixed Workload
- How to partition processing time?
  - Response times
  - Additional performance counters



# Estimation Approaches

## Utilization Approach

Data Collection

Demand Estimation

Modeling Assumptions  
(scheduling, service distribution)

Model Solution

## Response Time Approach

Data Collection

Modeling Assumptions  
(scheduling, service distribution)

Demand Estimation

Model Solution

# Linear Regression

## ■ Linear model (based on Utilization Law)

- $U_i = \sum_c^N X_{i,c} \cdot D_{i,c} + U_0$

## ■ Example:

$$0.54 = 3 * D_{i,1} + \dots + 8 * D_{i,n}$$

$$0.72 = 9 * D_{i,1} + \dots + 4 * D_{i,n}$$

$$0.33 = 2 * D_{i,1} + \dots + 9 * D_{i,n}$$

$$\dots = \dots$$

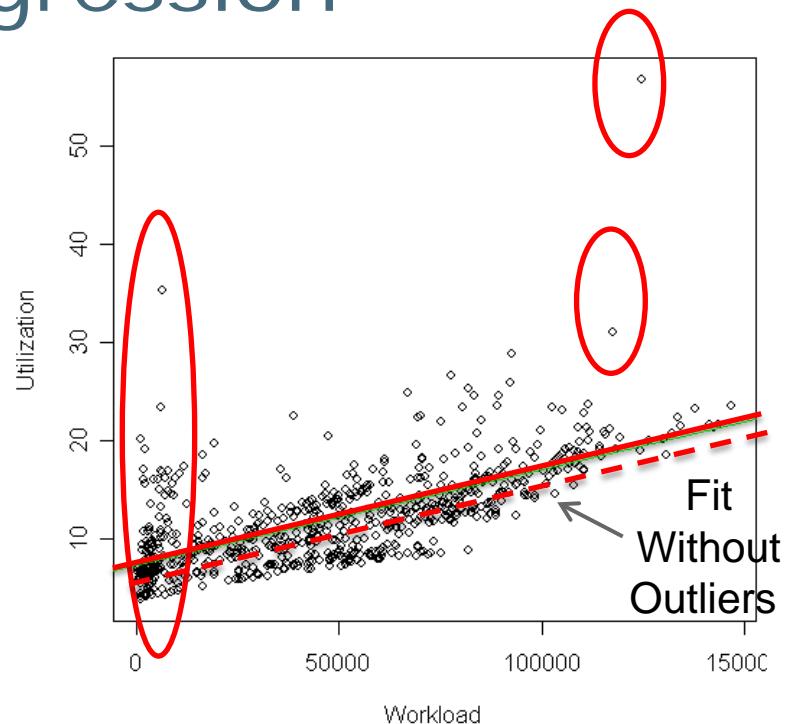
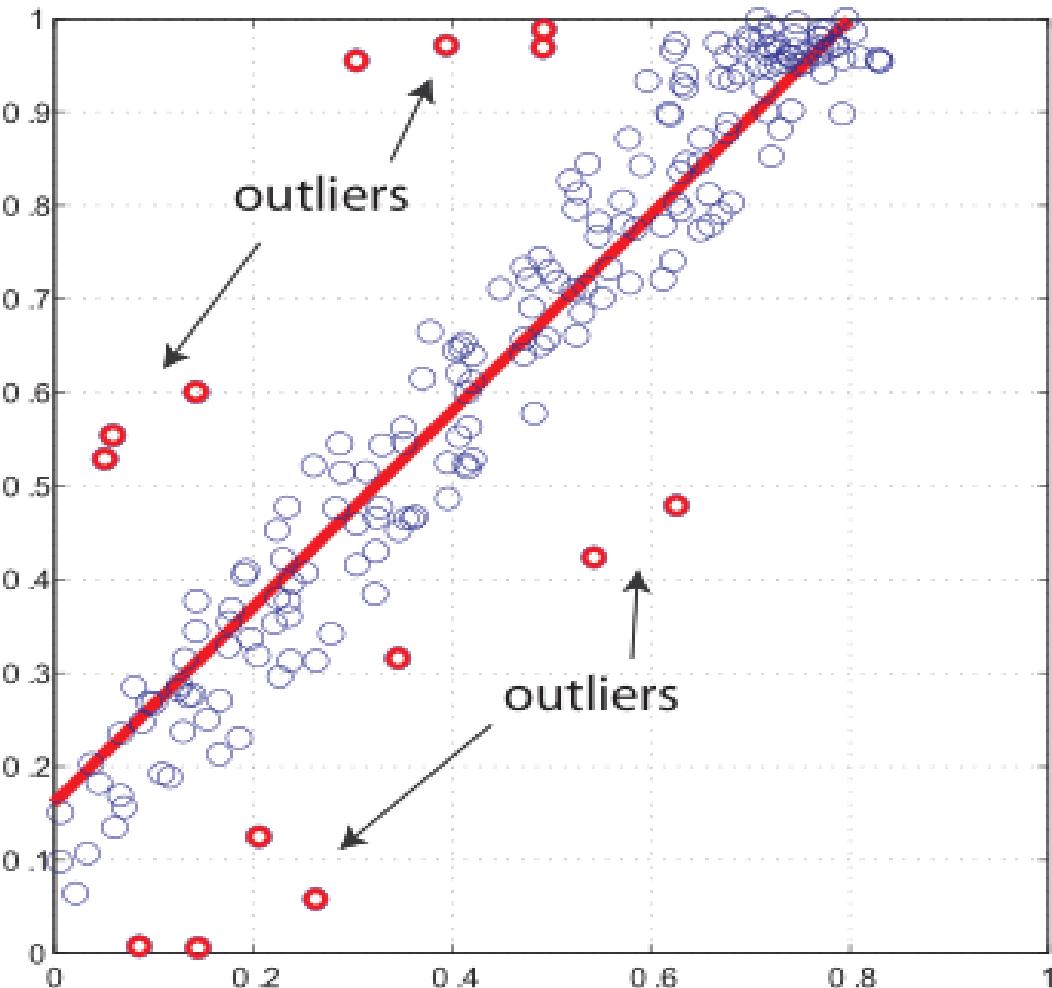
## ■ At least $m > n$ observations required

## ■ Alternative solutions:

- Least-Squares Regression
- Least Absolute Differences Regression
- ...

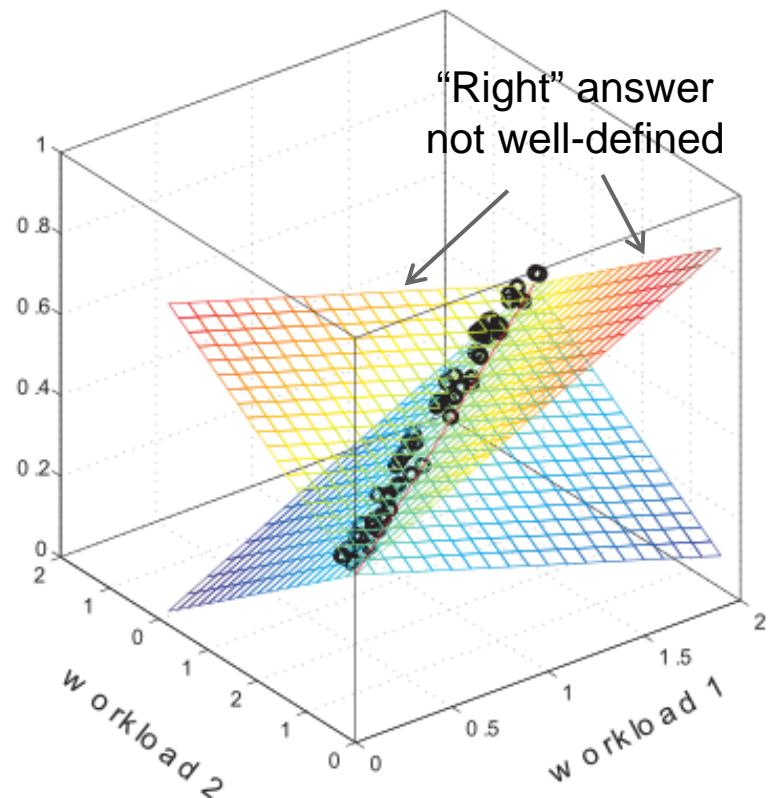
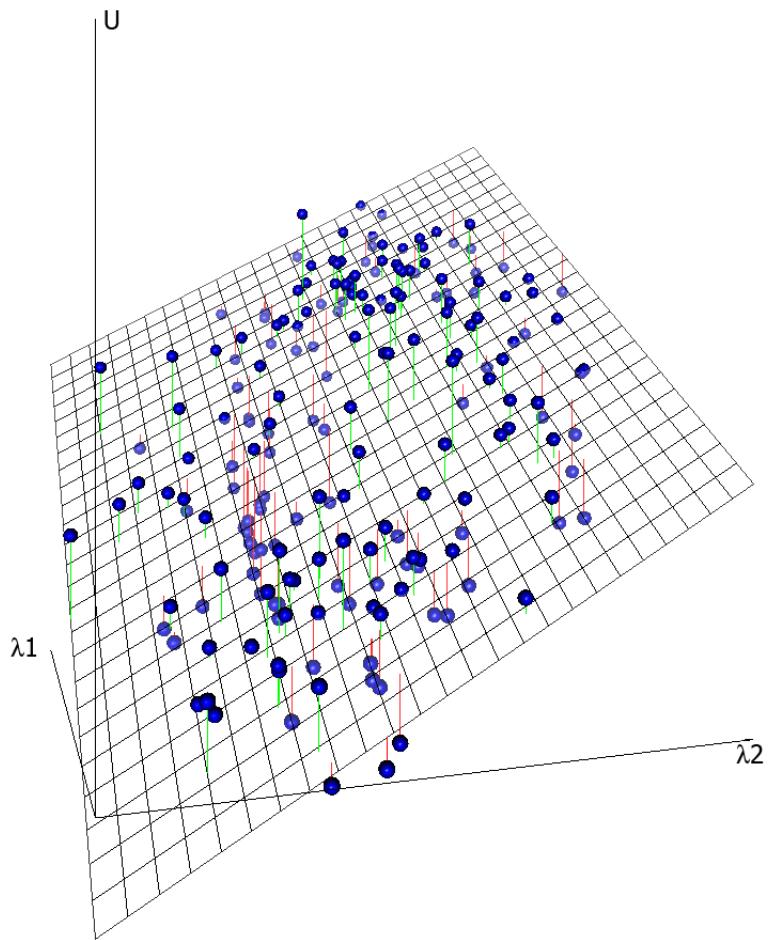
# Example: Outliers

- Outliers can bias the regression



# Example: Collinearities

- e.g.: logins proportional to logouts



# Other Approaches

## ■ Robust regression

- Least Absolute Differences: Zhang et al. (2007)
- Least Trimmed Squares: Casale et al. (2008)

## ■ Machine-learning

- Clusterwise linear regression: Cremonesi et al. (2010)
- Pattern matching: Cremonesi et al. (2014)

# Utilization Approaches

## ■ Utilization-based approaches

### ■ Advantages

- Only utilization and throughput data required
- Minimal assumptions:
  - Any scheduling strategy
  - Any interarrival distribution
  - (Any service time distribution)

### ■ Disadvantages

- Robustness
- Amount of data

# Response Time Approaches

## ■ Assumptions

- Single queue: closed-form solution exists
- Queueing network: product-form

## ■ Response time equations depend on

- Scheduling strategy
- Service distribution
- Interarrival time distribution

*If  $M/G/1$  with PS or LCFS  
or  $M/M/1$  with FCFS and class-independent service  
times, then  $R = \frac{D}{1-U}$*

## ■ Assumptions:

- Variables  $\mathbf{D} = (D_{1,1}, \dots, D_{1,n}, \dots, D_{i,1}, \dots, D_{i,n})$
- Queueing Network QN  $\rightarrow \mathbf{z} = f(\mathbf{D})$
- Observation data  $\tilde{\mathbf{z}}$

## ■ Optimization Problem:

- $\min_{\mathbf{D}} \|\mathbf{z} - \tilde{\mathbf{z}}\|$  ← arbitrary norm
- $\mathbf{D}$  may be subject to certain constraints

# Examples

## ■ Menascé (2008):

Squared response time error    Product-form solution (non-linear!)

$$\min \sum_{c=1}^C (R_c - \tilde{R}_c)^2 \quad \text{with} \quad R_c = \sum_{i=1}^I \frac{D_{i,c}}{1 - \sum_{d=1}^C \lambda_{i,d} D_{i,d}}$$

subject to  $D_{i,c} \geq 0 \quad \forall i, c$  and  $\sum_{c=1}^C \lambda_{i,c} D_{i,c} < 1 \quad \forall i.$

Constrained to valid solutions

## ■ Liu et al. (2006):

$$\min \sum_{c=1}^C p_c (R_c - \tilde{R}_c)^2 + \sum_{i=1}^I (U_i - \tilde{U}_i)^2$$

# Kalman Filter

## ■ Dynamical system

### ■ State model:

$$X_k = F_{k-1}X_{k-1} + B_{k-1}u_{k-1} + w_{k-1}$$

next state    previous state    controlled input    uncorrelated noise

### ■ Observation model:

$$Z_k = H_k X_k + v_k$$

observations    observation noise

## ■ Filter

$$z_k, z_{k-1}, z_{k-2}, \dots, z_1 \rightarrow \hat{X}_k \sim N(\hat{x}_k, \hat{P}_k)$$

time-series of observations

estimated mean value    estimated covariance

# Applied to Demand Estimation

- State vector  $X_k = D$
- Constant state model  $X_k = X_{k-1} + w_{k-1}$
- Observation model (e.g., Kumar et al. 2009)

$$\begin{pmatrix} R_1 \\ \vdots \\ U \end{pmatrix} = \begin{pmatrix} D_1 \\ \frac{D_1}{1-U} \\ \vdots \\ X_1 \cdot D_1 + \dots + X_C \cdot D_C \end{pmatrix}$$

- Other observation models are possible (e.g., Zheng et al. 2008, Wang et al. 2012)

## Elsevier PEVA, October 2015.

### Evaluating Approaches to Resource Demand Estimation

Simon Spinner<sup>a,\*</sup>, Giuliano Casale<sup>b</sup>, Fabian Brosig<sup>a</sup>, Samuel Kounov<sup>a</sup>

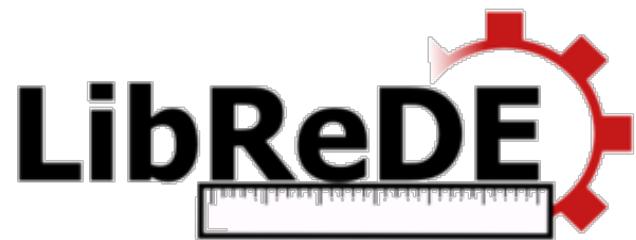
<sup>a</sup>*University of Würzburg, Am Hubland, Würzburg, Germany*

<sup>b</sup>*Imperial College London, Department of Computing, SW7 2AZ, UK*

---

#### Abstract

Resource demands are a key parameter of stochastic performance models that needs to be determined when performing a quantitative performance analysis of a system. However, the direct measurement of resource demands is not



<http://descartes.tools/librede>  
Eclipse Public License (EPL)

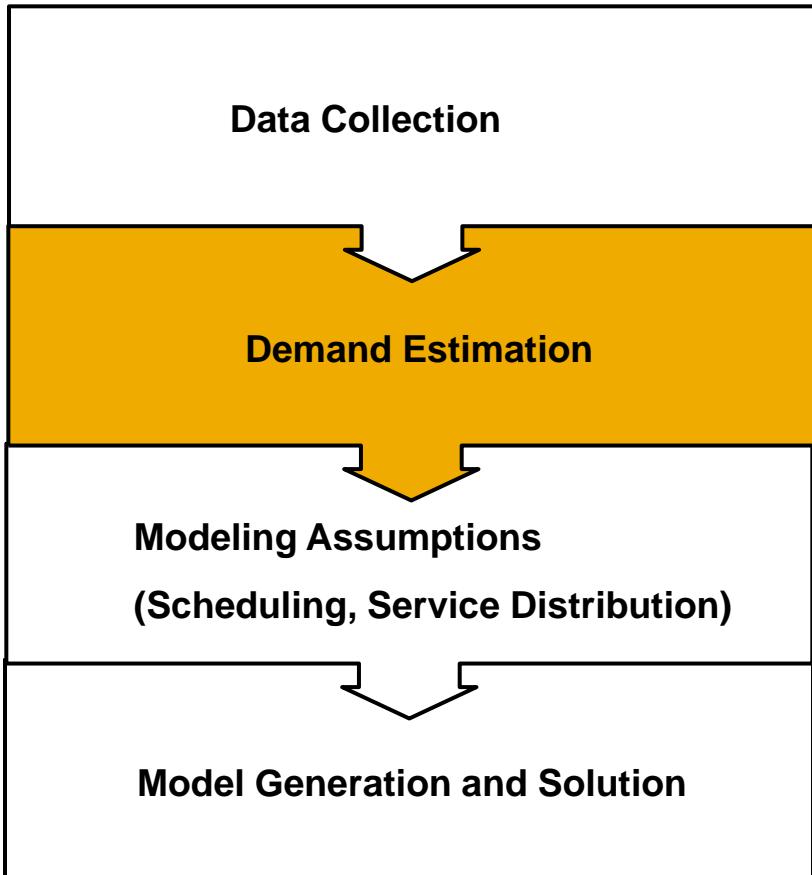
vmware®



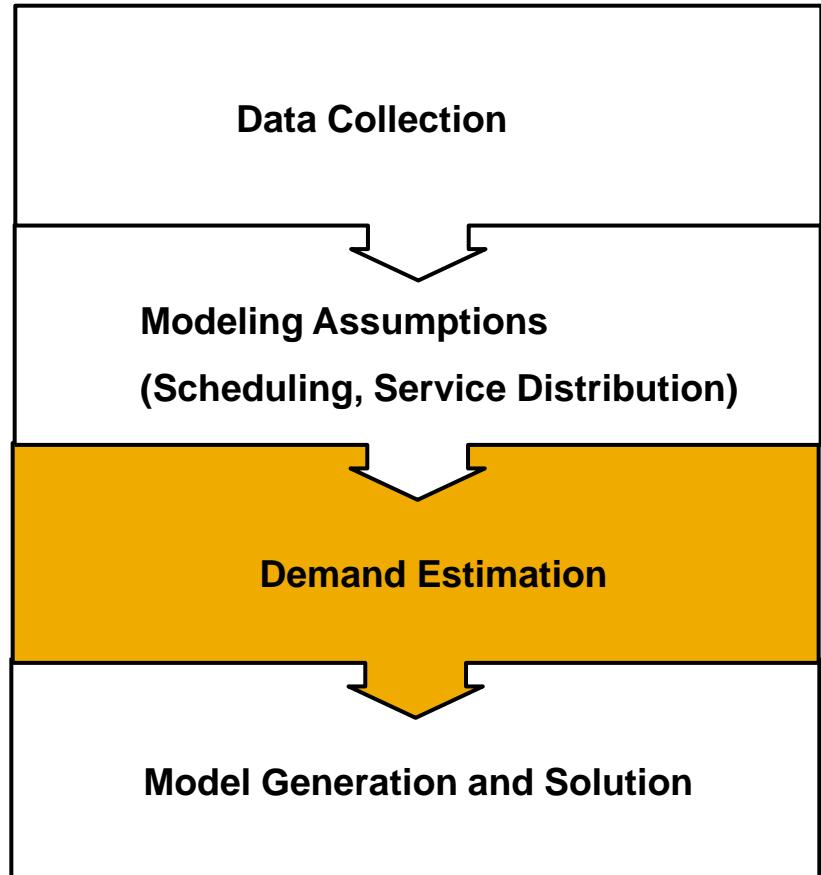
# Response Time Based Estimation

*Joint work with S. Kraft and S. Pacheco-Sánchez (SAP Belfast,  
UK) and Juan F. Pérez (U. Melbourne, Australia)*

## Utilization Approach

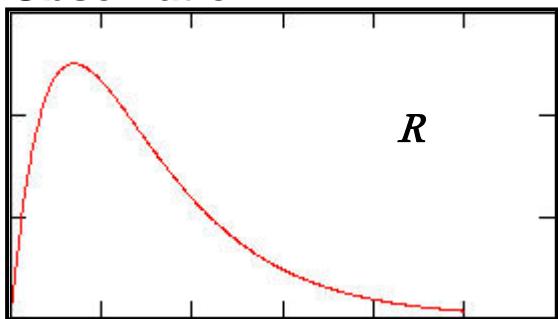


## Response Time Approach



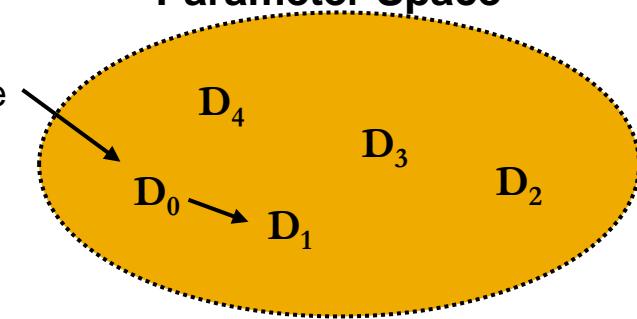
## Estimate demand $D$ from response time $R$

**Observation**



1. For each observed  $R$  sample
2. Draw  $D$  from parameter space
3. Compute likelihood  $P[R|D]$
4. Move in parameter space to maximize  $P[R|D]$

**Parameter Space**

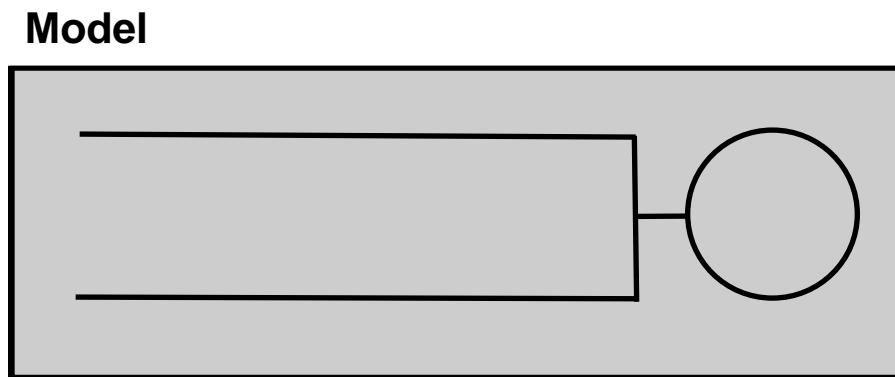
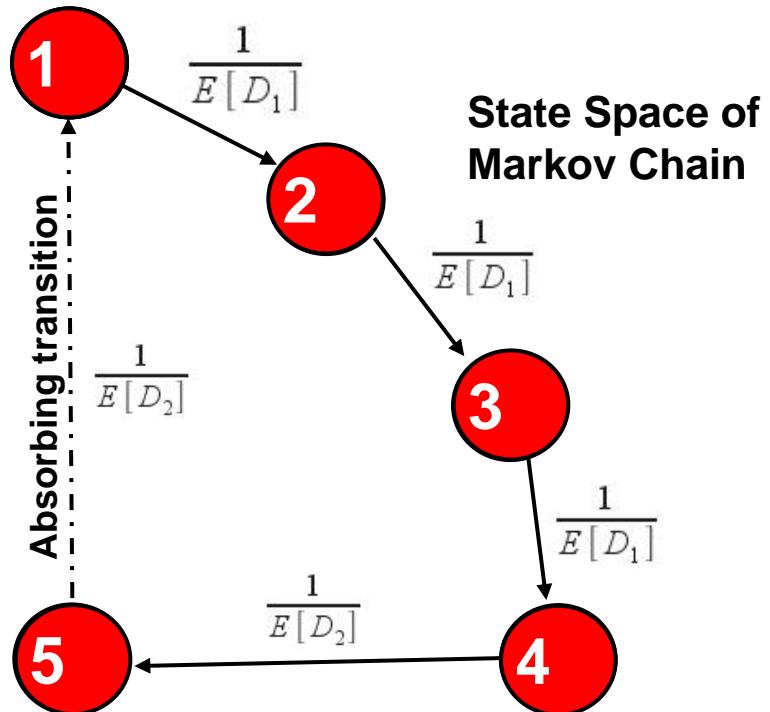


## We investigate the likelihood function in

- First-Come First-Served (FCFS) queues
  - e.g., admission control, disk drive buffers, ...
- Processor Sharing (PS) queues
  - e.g., CPUs, bandwidth sharing, ...

# RT Likelihood in FCFS queues

- Model response time using absorbing CTMCs
- Under FCFS, future arrivals do not affect response time distribution of the tagged job



- Probability of being absorbed by time t into a give CTMC state
- Well-understood: PH-type distributed

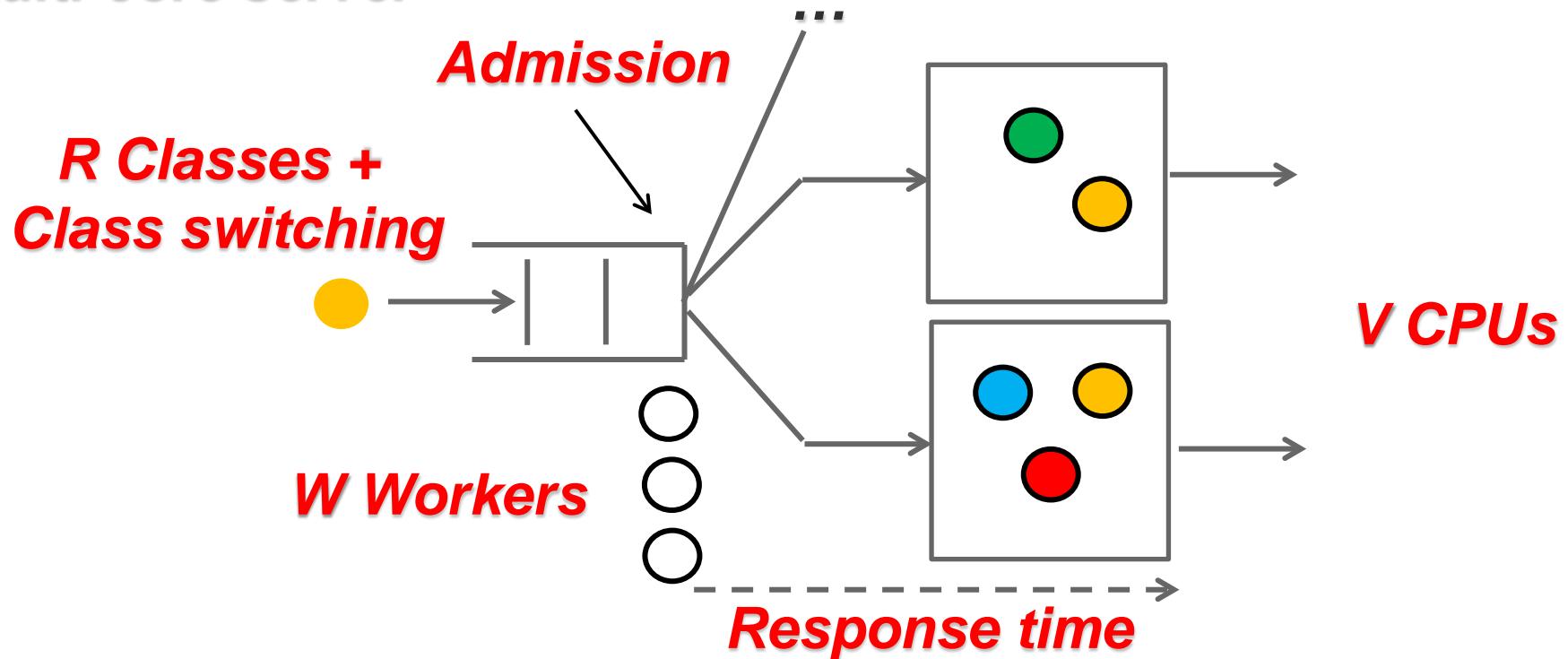
FCFS Example:  $\Pr[R_{meas}^i | E[D_1], \dots, E[D_K]] = [1, 0, \dots, 0] e^{\mathbf{T} R_{meas}^i} (-\mathbf{T}) \mathbf{1}$

$$\mu_k = 1/E[D_k] \quad \mathbf{T} = \begin{bmatrix} -\mu_1 & \mu_1 & & & \\ & -\mu_1 & \mu_1 & & \\ & & -\mu_1 & \mu_1 & \\ & & & -\mu_2 & \mu_2 \\ & & & & -\mu_2 \end{bmatrix} \quad \begin{array}{l} \text{Class 1 arrival} \\ \text{Backlog seen upon arrival} \end{array}$$

ML Problem (K classes)  $\max \sum_{i=1}^I \log \Pr[R_{meas}^i | E[D_1], \dots, E[D_K]]$

# Assumptions

## Multi-core server



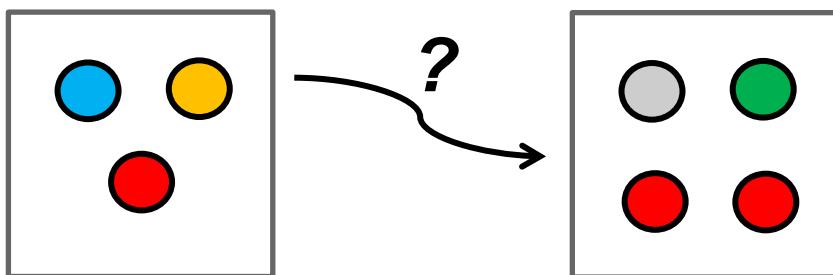
- Monitoring dataset
  - Active mix: (1 blue circle, 2 yellow circles, 1 red circle, 1 green circle, 0 purple circles)
  - Admission state (mix) and response times

## ■ Class switching probabilities

- Users submit requests cyclically
- Requests issued change class over time

## ■ Closed class-switching queueing model

- $V$  CPUs,  $R$  classes  $\rightarrow O(V^{2R})$  states
- Inference of trajectory too complex



- CI: Complete information (baseline)
  - $V=1$  CPU: full state trajectory
  - $V>1$  CPUs: no individual CPU states
  - We split demand proportionally, taking into account the active workers
- PI: Partial information
  - Sample admission state and response time
  - Mean throughput is assumed known

- $V=1$  CPU

- Full demand distribution

$$d_{j,r} = \sum_{i=1}^{I-1} \frac{(t_{j,i+1} - t_{j,i})}{n(t_{j,i}^+)} \quad \begin{matrix} \text{\color{red} Request} \\ \text{\color{red} Runtime} \end{matrix}$$

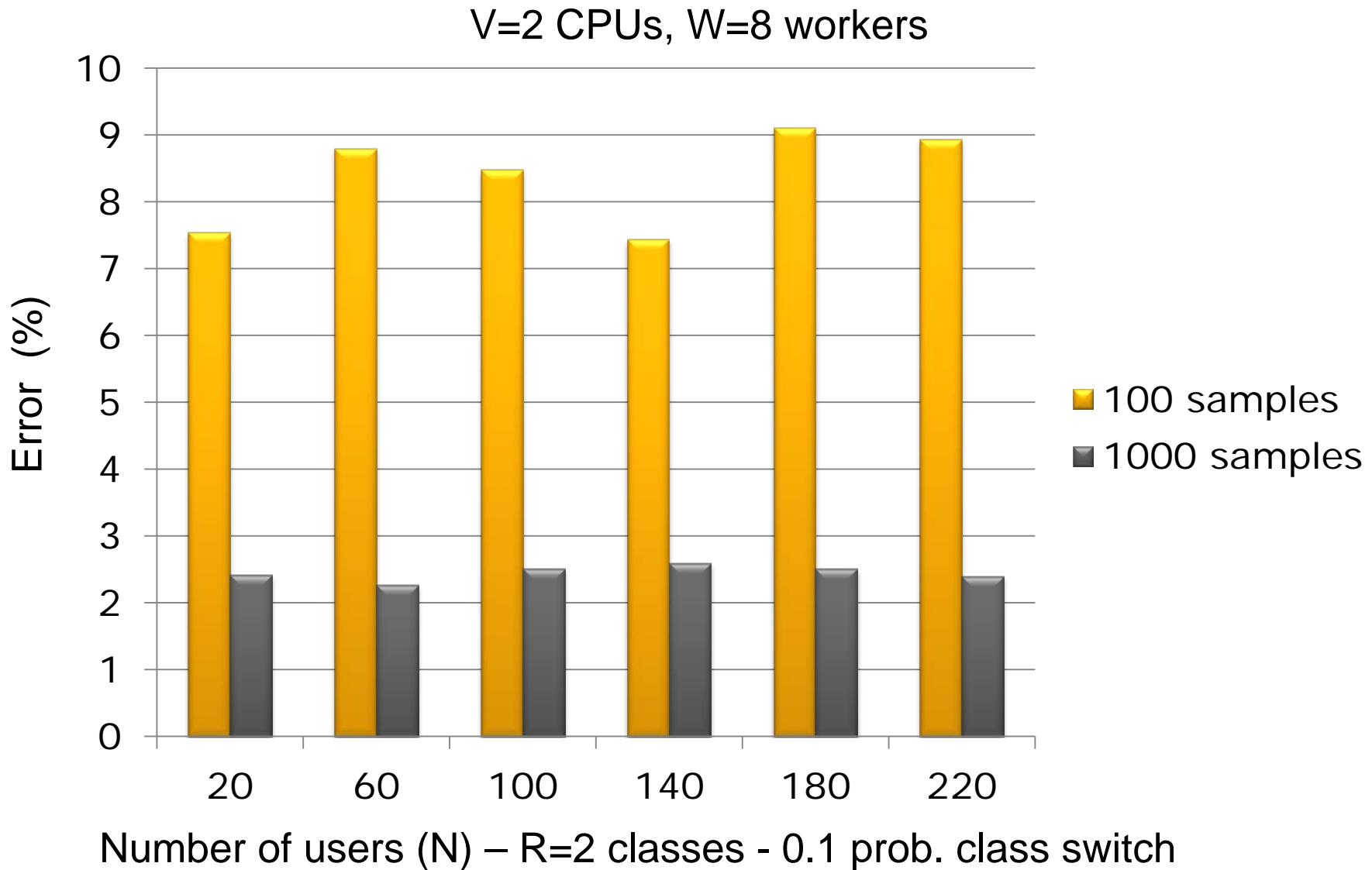
**Demand**  $\rightarrow$  **Request  $j$**   $\rightarrow$  **Class  $r$**

**Active workers**  $\leftarrow$   $n(t_{j,i}^+)$

- $V>1$  CPU

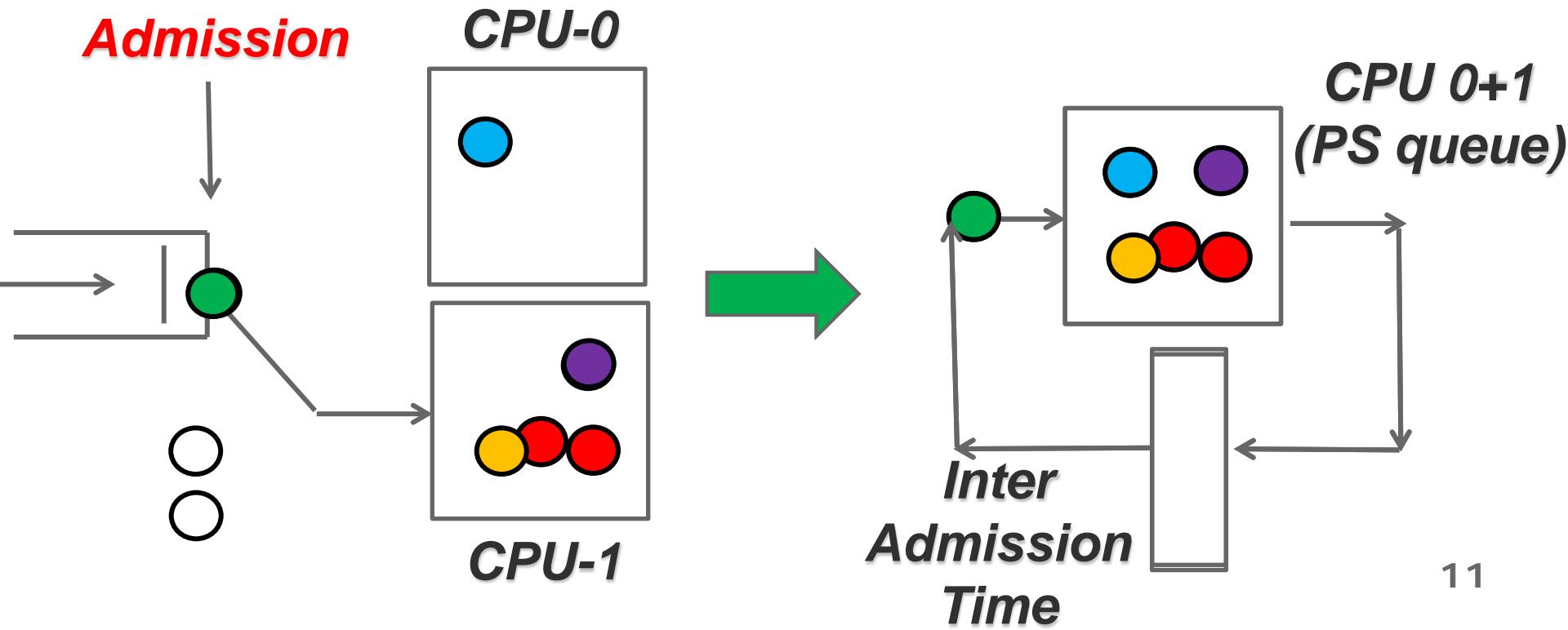
**Scale by Active CPUs**

$$d_{j,r} = \sum_{i=1}^{I-1} \frac{(t_{j,i+1} - t_{j,i}) \min(n(t_{j,i}^+), V)}{n(t_{j,i}^+)} \quad \begin{matrix} \downarrow \\ \text{\color{red} Scale by Active CPUs} \end{matrix}$$



# PI: Approximation

- CI requires very detailed measurements
- Closed queueing network model
  - Assume a fixed mix as seen upon job arrival
  - No class switching ( $\rightarrow$  tractable)
  - Model can estimate response time of arriving job



# RPS: Regression Approach

## ■ V=1 CPU

- Model assumed in equilibrium

**Queue seen  
at admission  
(incl. arriving job)**

**Response  
Time  
Class r**

$$E[R_r] = E[D_r] E[\bar{Q}^A]$$



**Demand  
Class r**

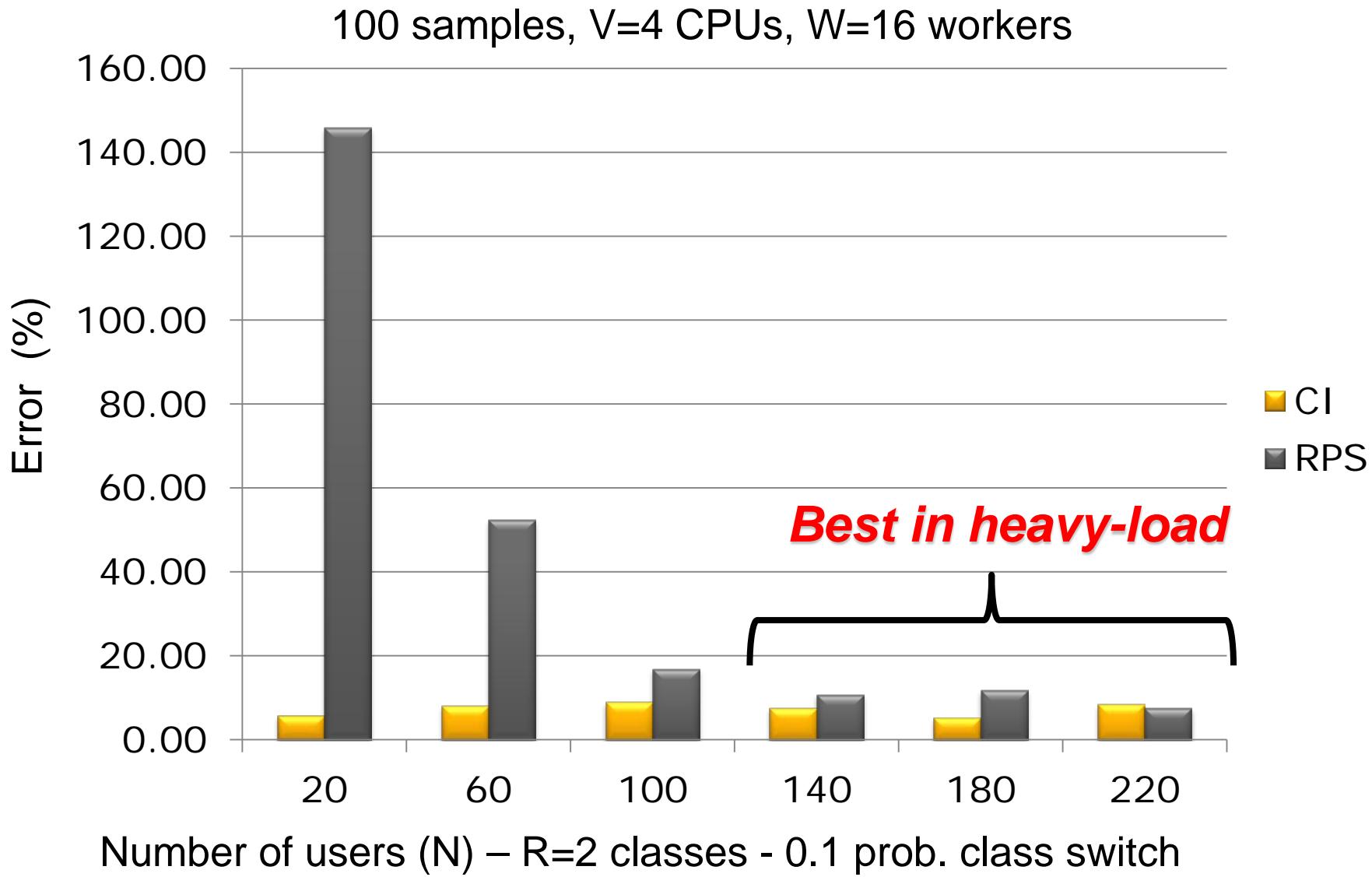
## ■ V>1 CPU

- Individual CPU state estimated

$$E[R_r] = E[D_r] E[\bar{Q}^A] / V,$$



**Average queue per CPU**



## ■ Maximum Likelihood Estimation (MLPS)

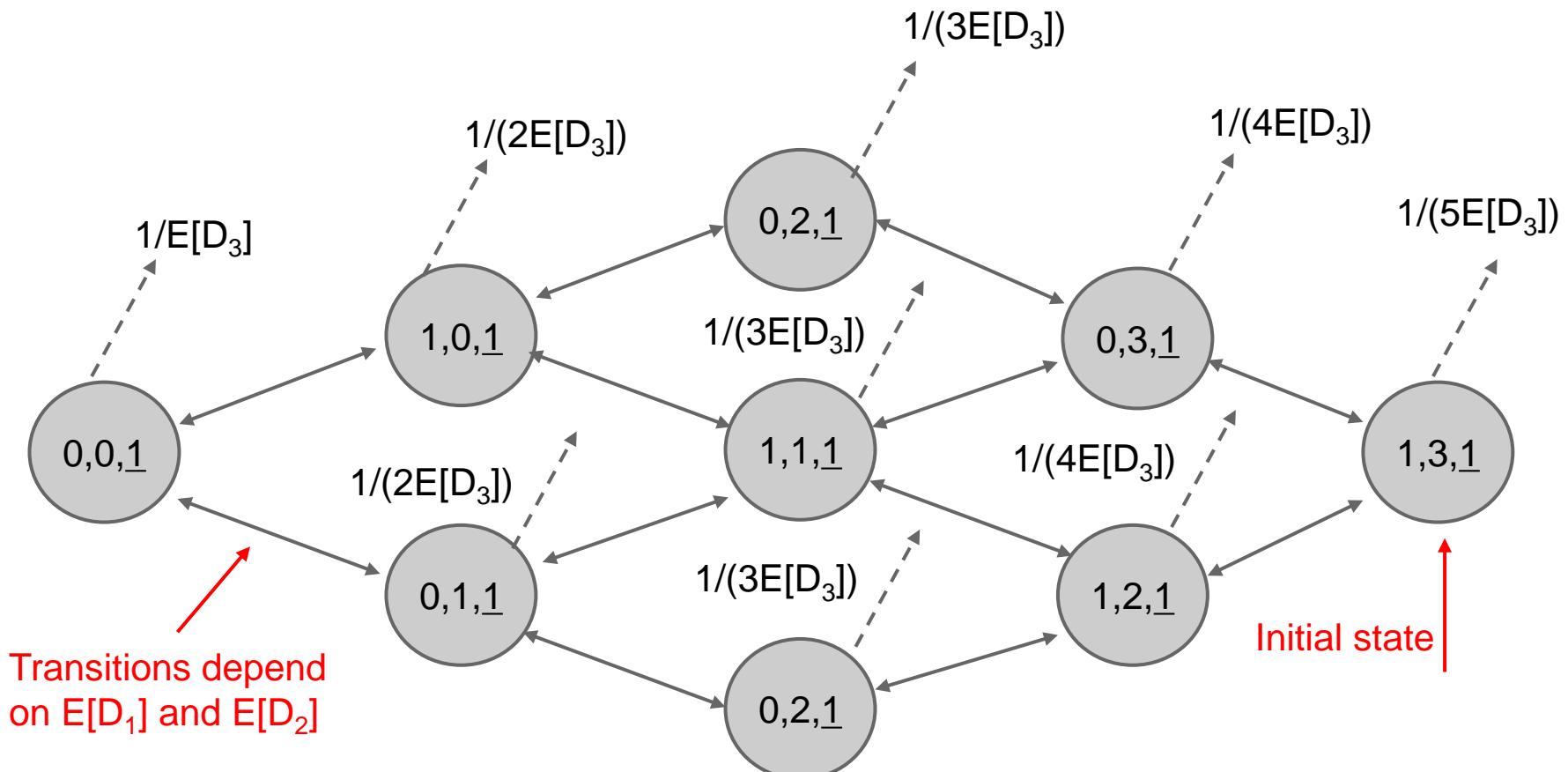
- Search over mean demand guesses
- Maximize likelihood of observed dataset

## ■ Response time likelihood

- Tagged customer method (**absorbing CTMC**)
- Initialized with state seen upon admission
- Mean demand guess → CTMC rates

# MLPS: Absorbing CTMC

- Job of class 3 arrives at system with  $V=1$  CPU
- Mix seen upon arrival: 1 job of class 1, 3 jobs of class 2



- We study the transient of this CTMC to obtain the response time distribution of the class-3 job

# MLPS: Absorbing CTMC

## ■ V=1 CPU

- Dataset:  $\{(n(t_i), r_i)\}_{i=1}^I$
- Likelihood function for each sample:

$$f(r_i|\mu) = \alpha(n(t_i)) \exp(T(n(t_i), \mu)r_i)t(n(t_i), \mu)$$

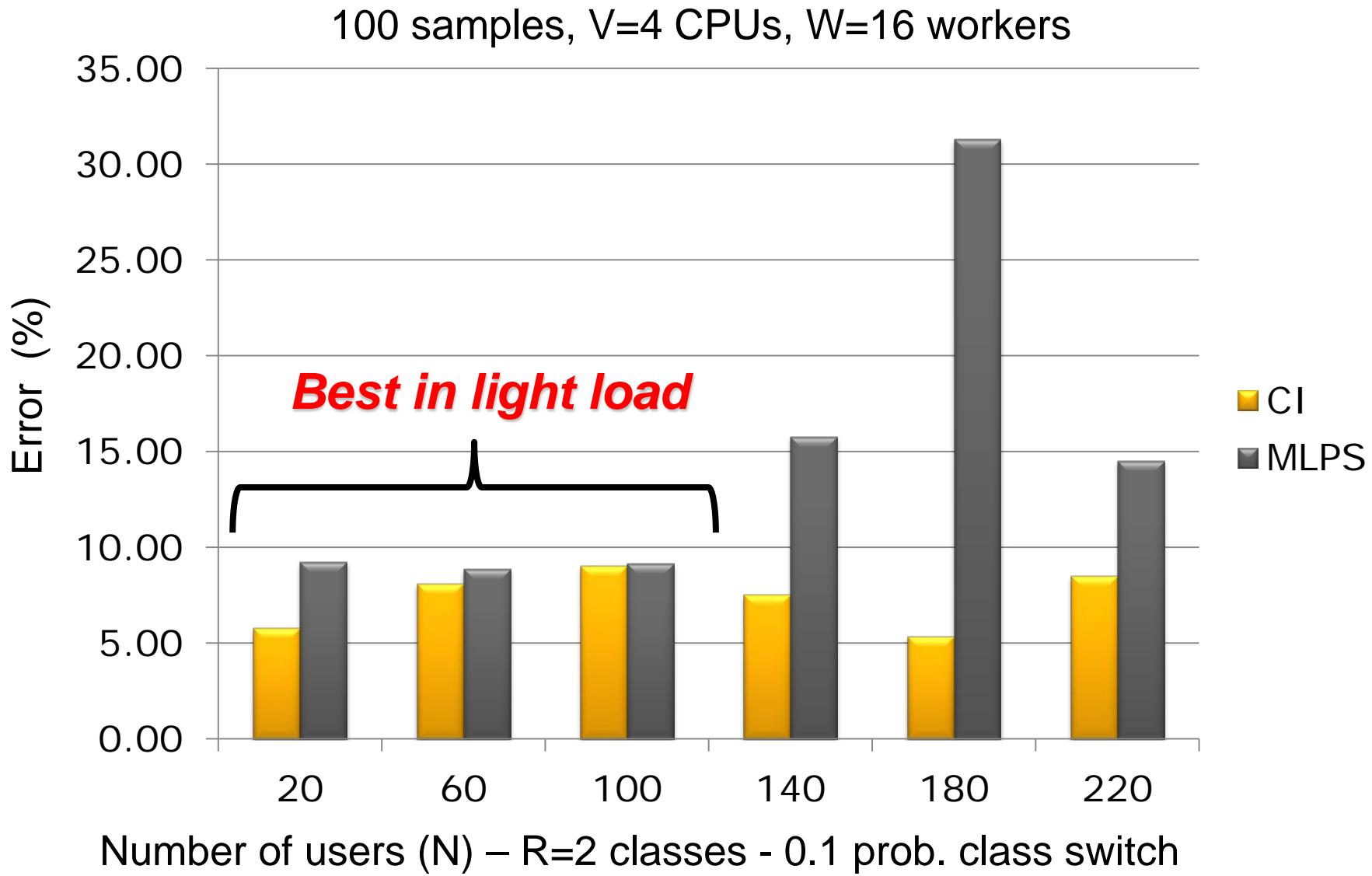
Diagram illustrating the components of the likelihood function:

- 1/demand** (red text) points to  $f(r_i|\mu)$ .
- CTMC generator** (red text) points to  $t(n(t_i), \mu)$ .
- init with state at admission** (red text) points to  $n(t_i)$ .
- trajectory in  $r_i$ , sec** (red text) points to  $r_i$ .
- completion rates** (red text) points to  $\alpha(n(t_i))$  and  $T(n(t_i), \mu)$ .

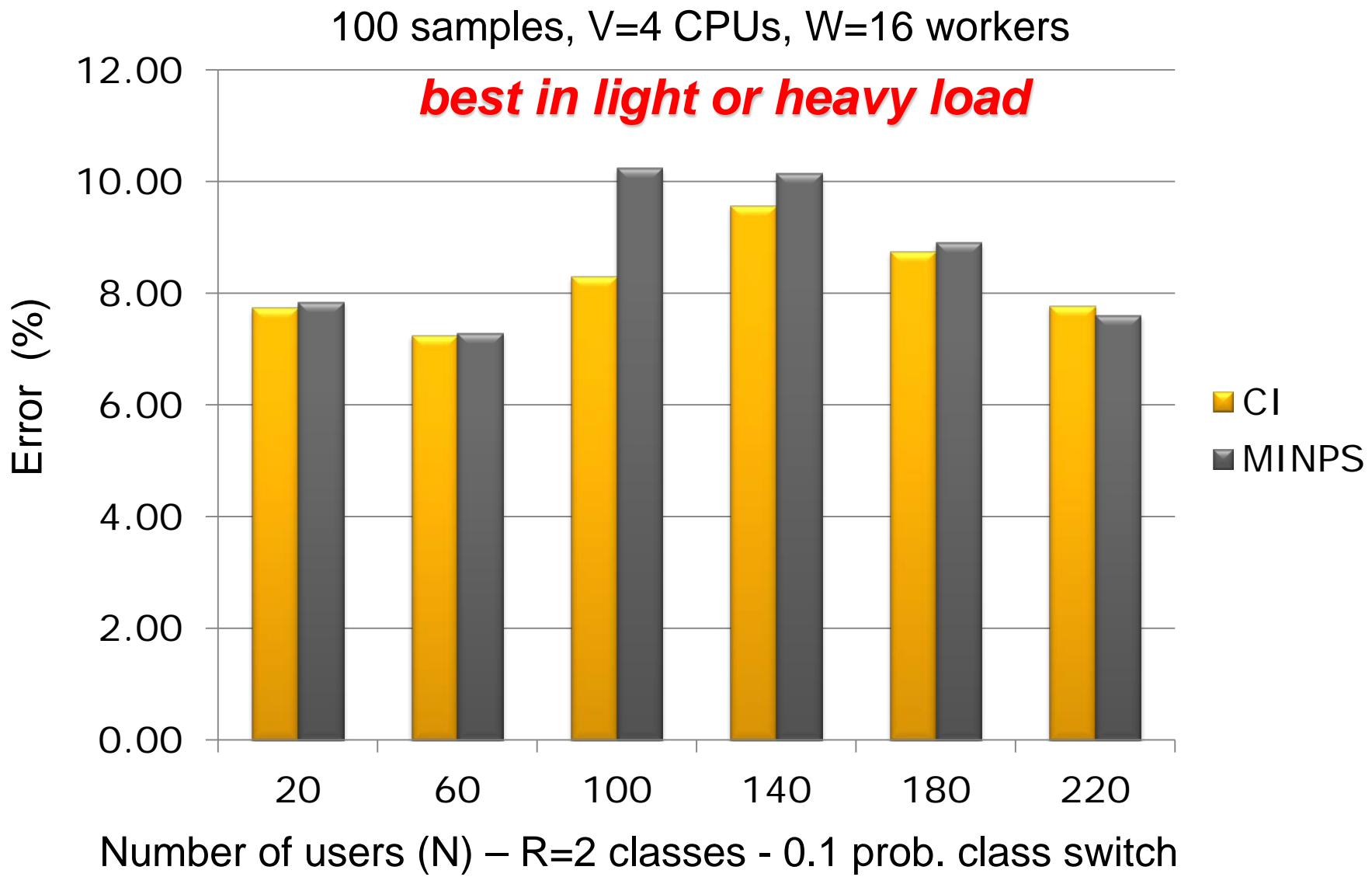
## ■ V>1 CPUs

- Load-dependent rates

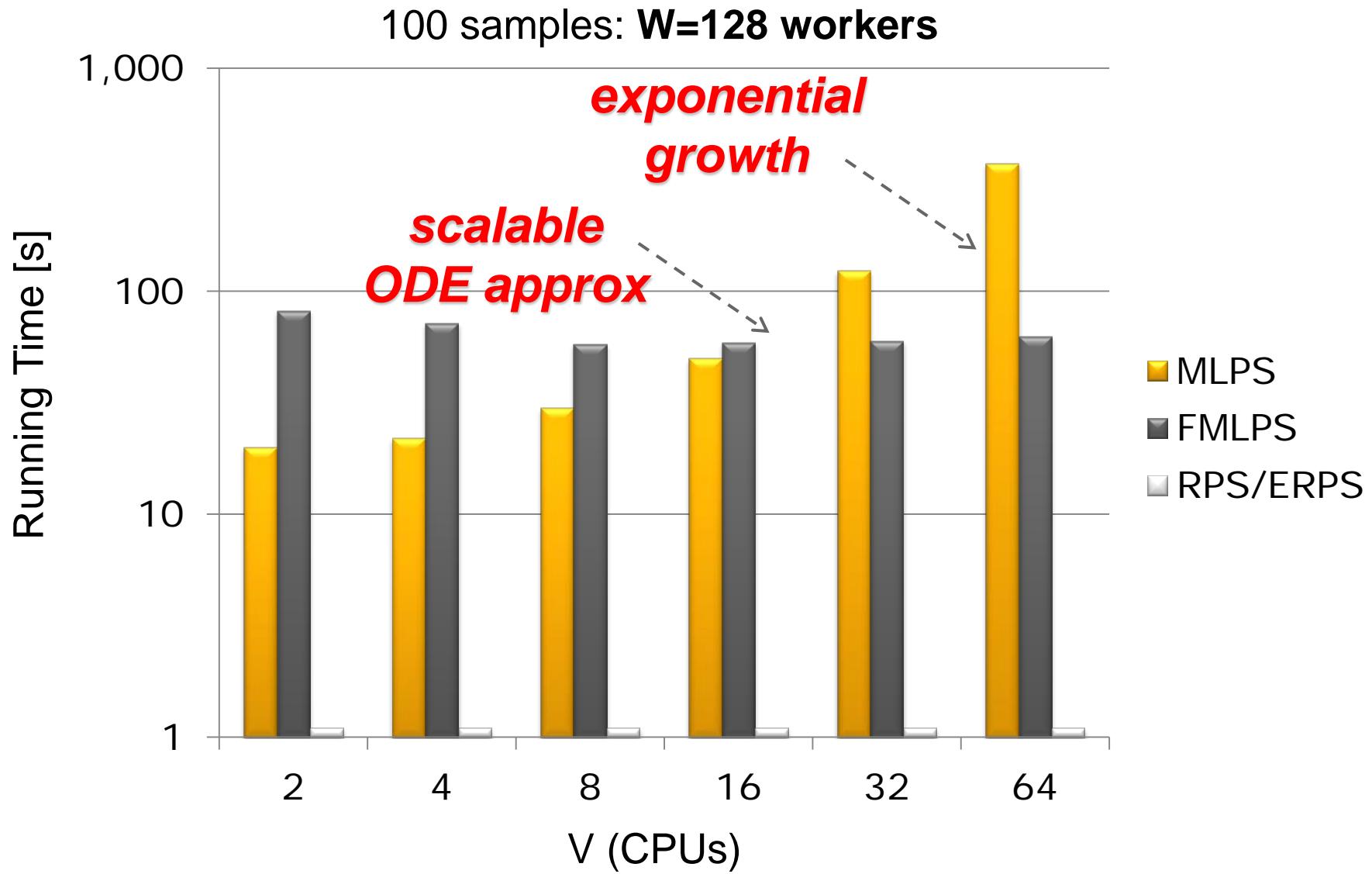
# MLPS: Results



# MinPS: $\min(D_{RPS}, D_{MLPS})$



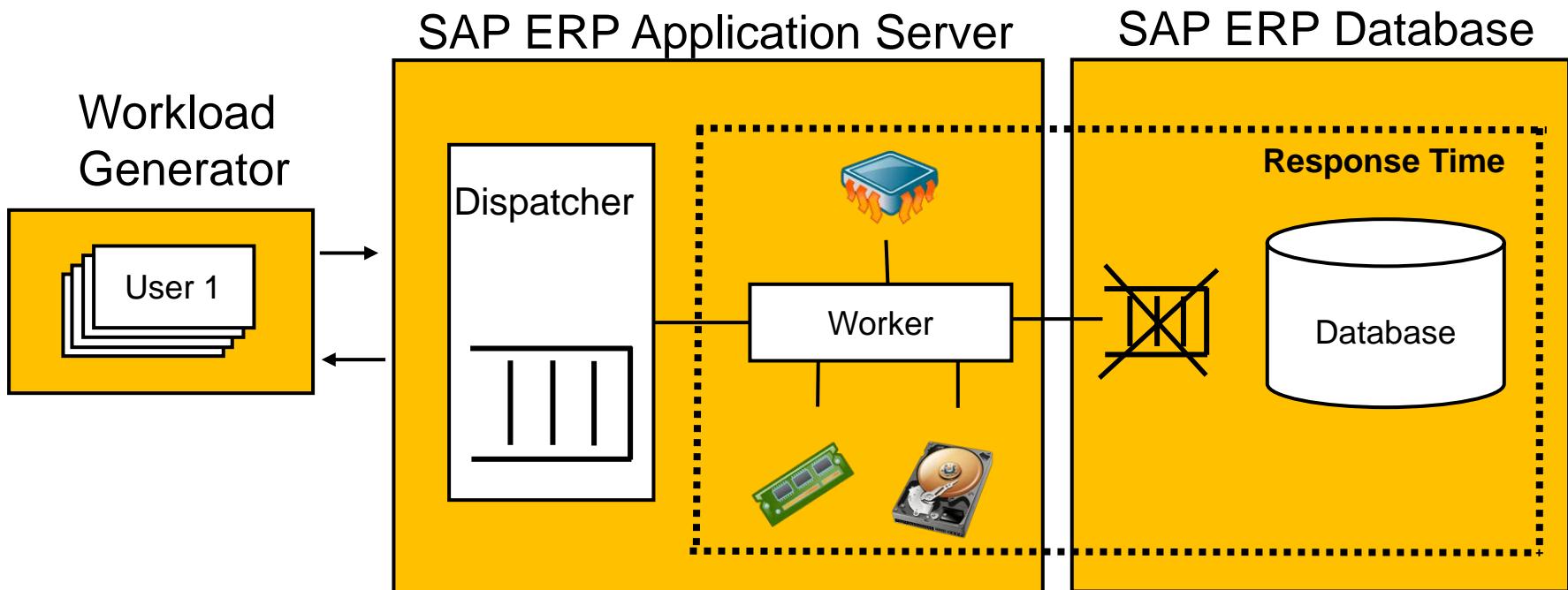
# MinPS: ODE Approximation



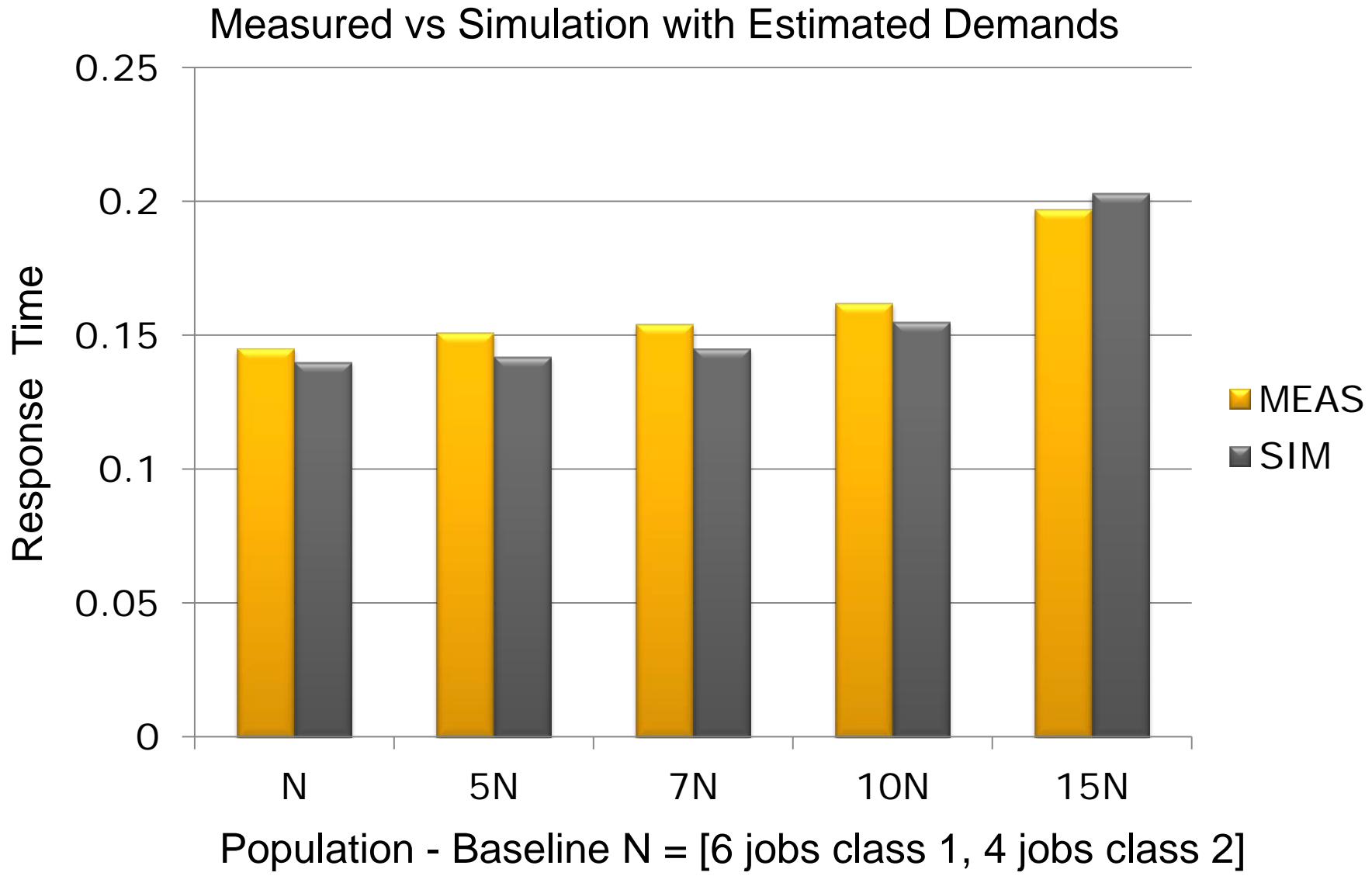
- Magnitude of class demands
  - 3 orders of difference: CI gap ~insensitive
- Class switch probability
  - High / Rare: CI gap ~insensitive
- Non-exponential service
  - low CV: CI gap weakly sensitive
  - high CV: CI gap ~insensitive for  $CV < 2$

# Case Study: SAP ERP

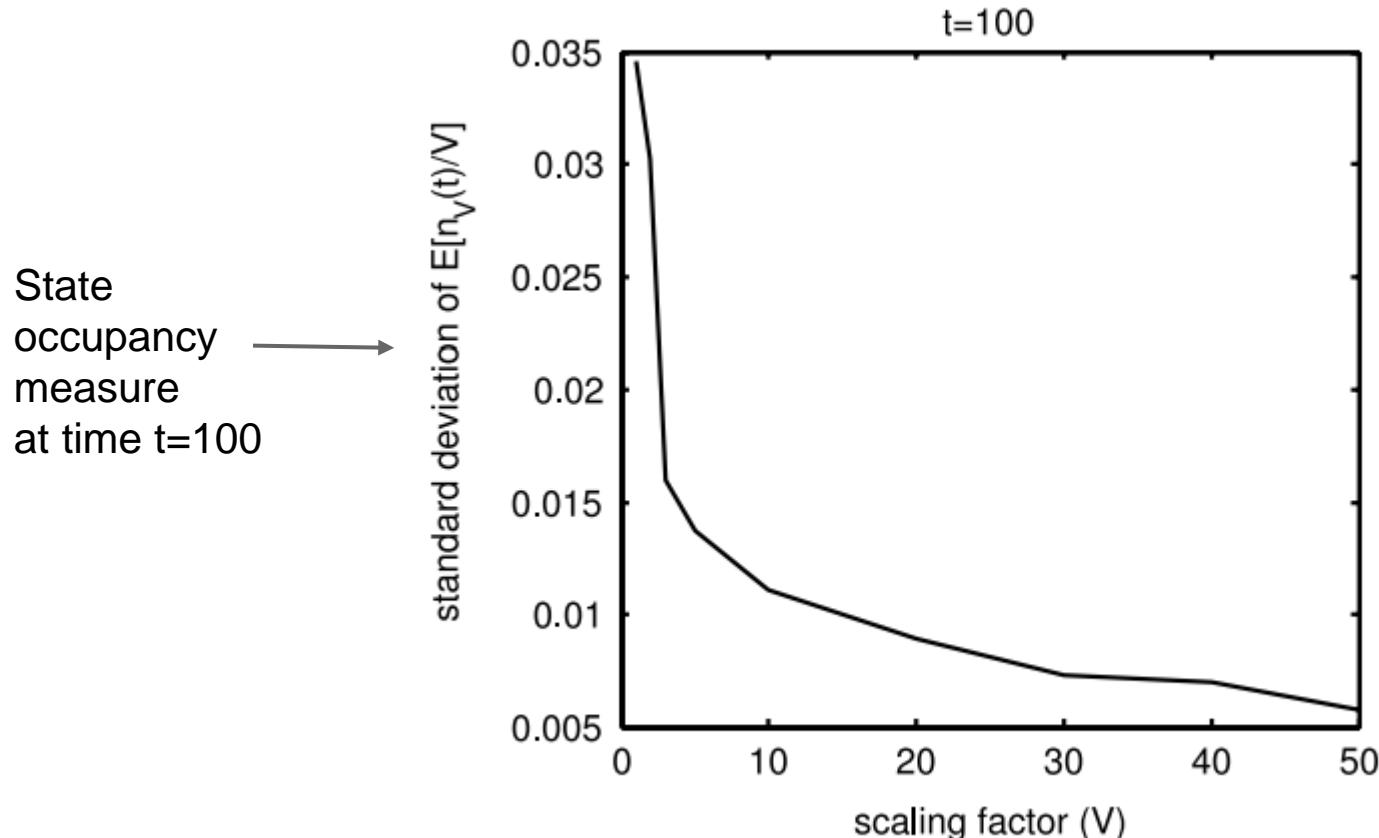
- 3-tier commercial application
  - Modified MLPS with **setup times**
  - Transactions grouped in R=2 classes



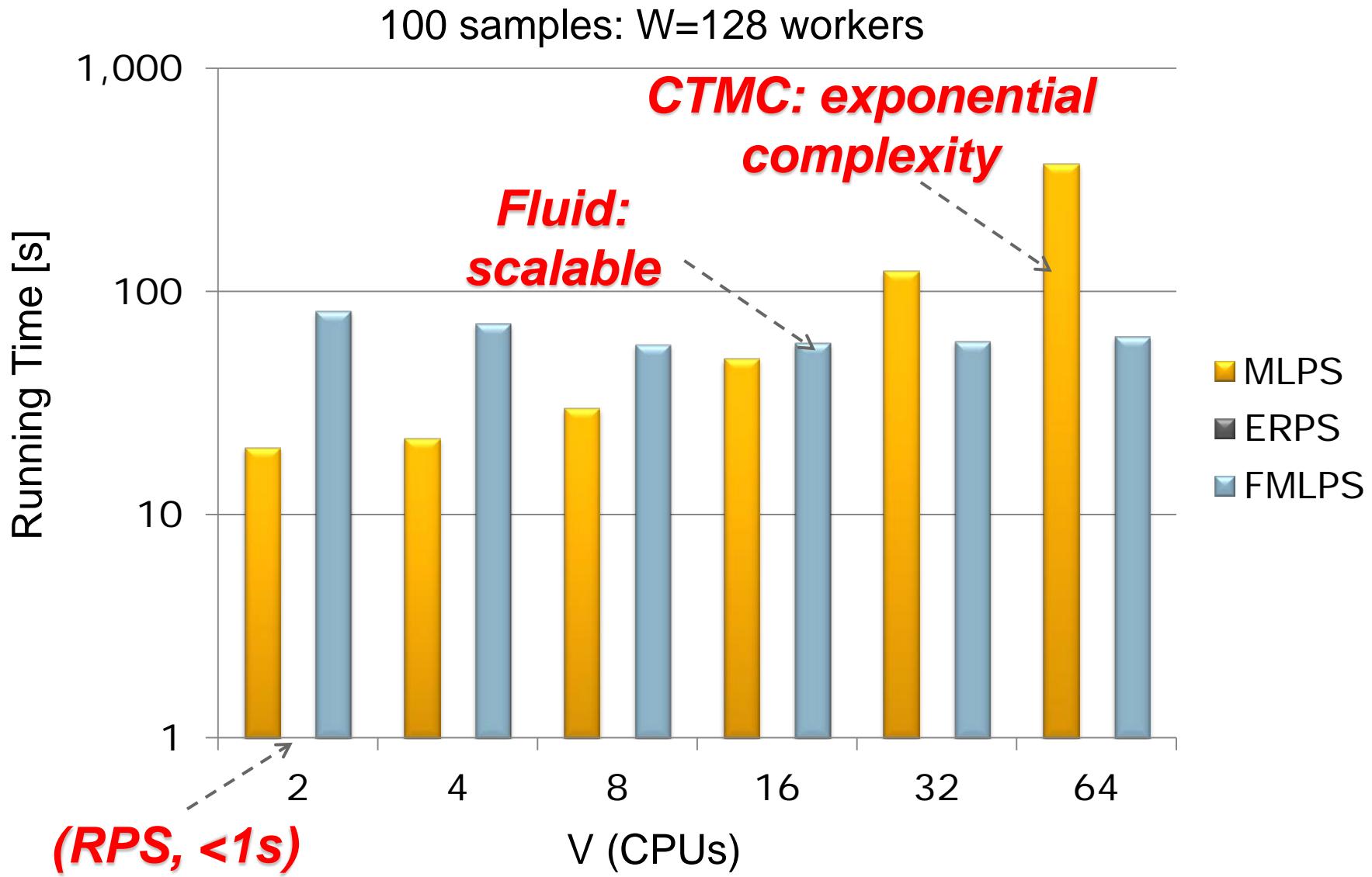
# SAP ERP Queueing Model



- Limit behaviour of the CTMC for growing rates and requests increasingly deterministic
  - $V$ =scale factor. Request mix is unchanged.
  - Limit behaviour can be modelled via ODEs

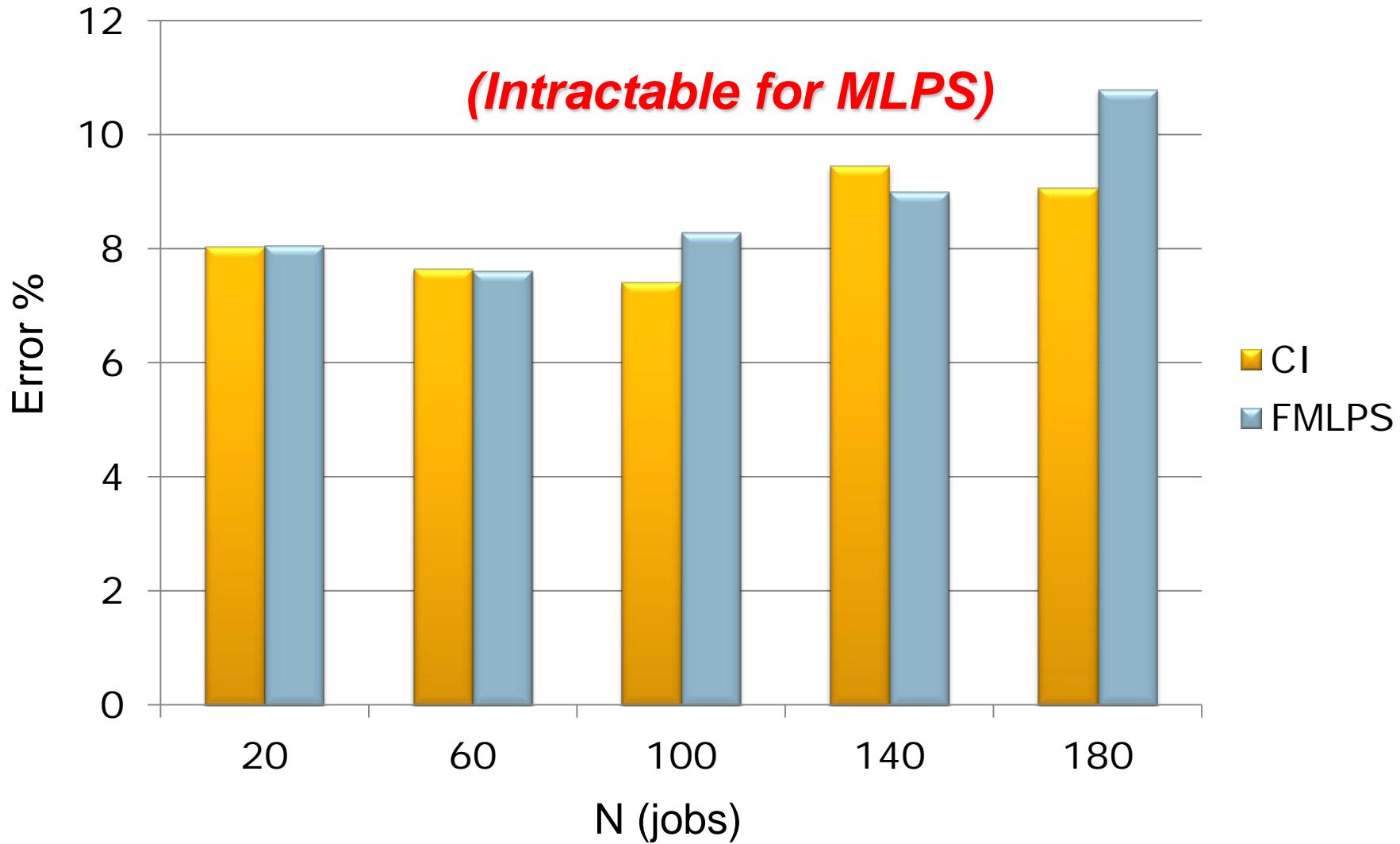


# Computational Costs



# FMLPS: SAP ERP Results

100 samples: V=8 CPUs, W=64 workers



# Queue-Length Based Estimation

# Gibbs Sampling (GQL)

- Monitor occupancy at all resources
  - Observations:  $\{\mathbf{n}^{(1)}, \mathbf{n}^{(2)}, \dots, \mathbf{n}^{(L)}\}$
  - Ill-posed, unless think times known
- Probabilistic model of distributed system

$$P(\mathbf{n}|\mathbf{D}) = \frac{Z_1^{n_{01}}}{n_{01}!} \cdots \frac{Z_R^{n_{0R}}}{n_{0R}!} \prod_{i=1}^V n_i! \prod_{r=1}^R \frac{D_{ir}^{n_{ir}}}{n_{ir}! G(\mathbf{D})}$$

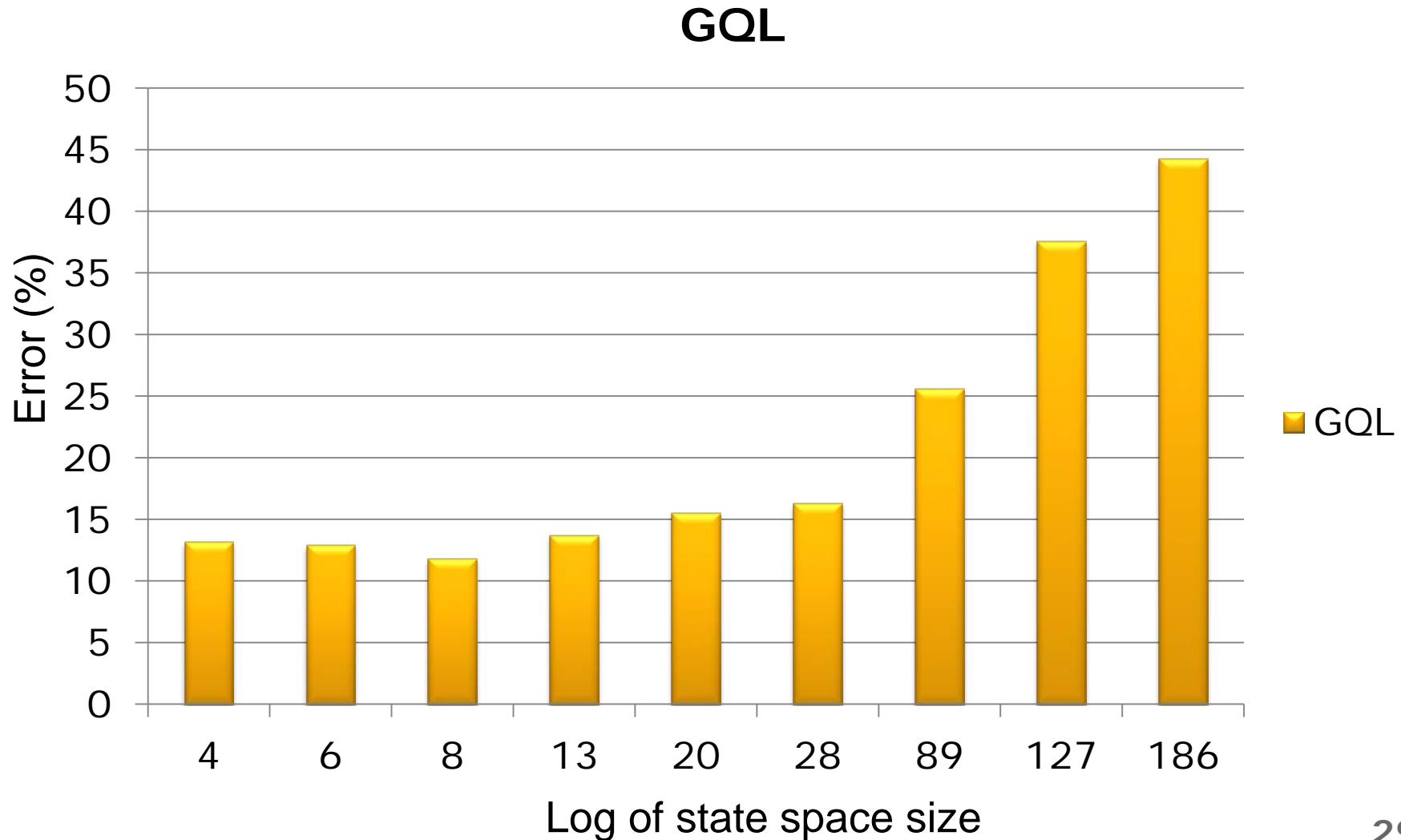
- Gibbs: iteratively sample posterior prior

$$\prod_{i=1}^L P(D_{ir}|\mathbf{D}_{(ir)}, \mathbf{n}^{(i)}) = \prod_{i=1}^L \frac{P(\mathbf{n}^{(i)}|\mathbf{D})P(\mathbf{D})}{\int_{D_{ir}} P(\mathbf{n}^{(i)}|\mathbf{D})P(\mathbf{D})dD_{ir}}$$

- Accurate estimates, error ~3%-7%

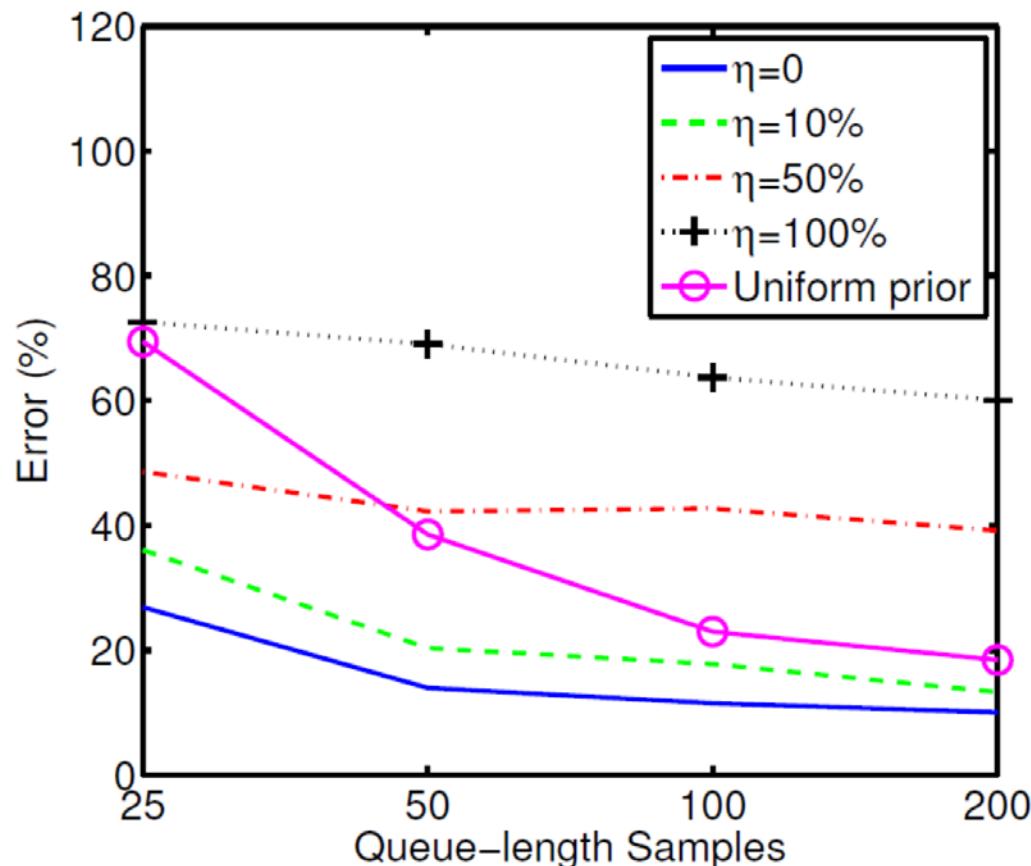
$R$	$V$	Time(s)	Average	C.I.
1	1	38	4.27%	4.91% - 3.64%
1	3	148	3.45%	3.75% - 3.16%
2	1	83	3.77%	4.17% - 3.37%
4	4(Ser)	1489	5.05%	5.22% - 4.88%
4	4(Pal)	1460	7.48%	7.16% - 7.79%

## ■ Increasing model sizes

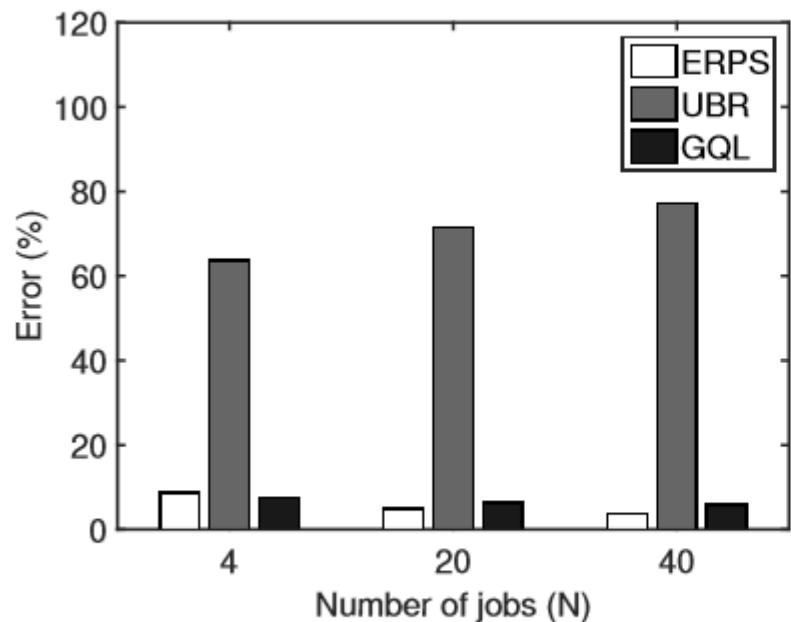


## ■ Dirichlet prior

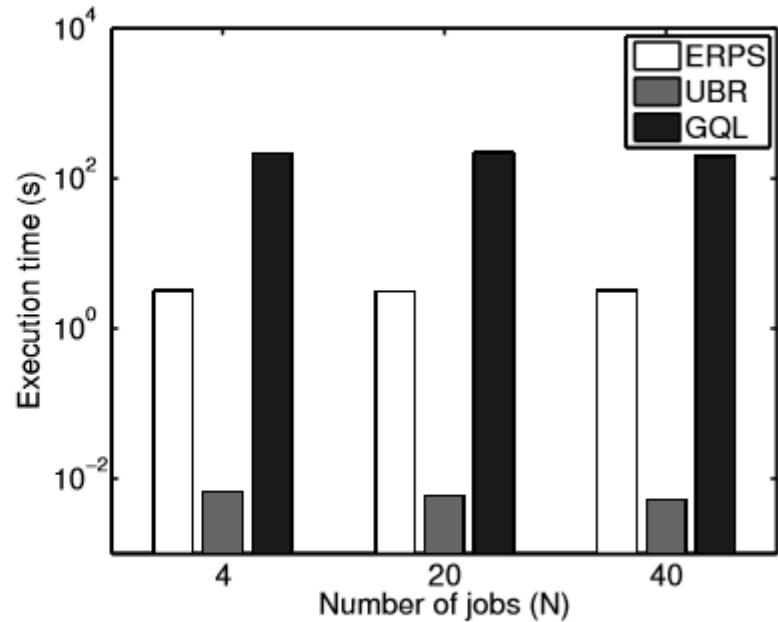
- all estimates converge unless the exact demand has very low probability prior



■ Accurate estimates, error ~3%-7%



(a) Estimation Error



(b) Execution time

# GQL vs Other MCMC Methods

- Far better convergence properties than Metropolis-Hastings and Slice sampling

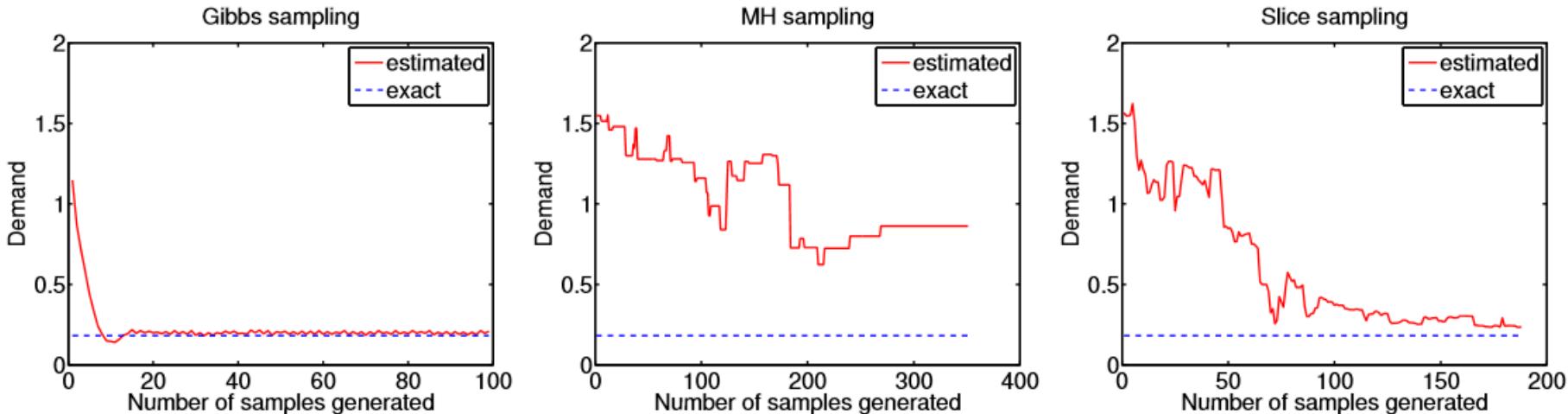


Fig. 4: Convergence plots

- About 13-15% error in estimating demands against cloud ERP data (Apache OFBiz)

- Based on Maximum Likelihood Estimation
- Works with mean queue length
- A simple approximation of the MLE:

- Consider the demand vector  $\theta^{bs}$  where

$$\theta_{ij}^{bs} = \frac{\tilde{Q}_{ij}(D)}{(N_j - \sum_{k=1}^M \tilde{Q}_{kj})} \frac{\theta_{0,j}}{(1 + \sum_{h=1}^R \tilde{Q}_{ih} - \tilde{Q}_{ij}(D)/N_j)}$$

- More details at tomorrow's talk!

# FG Tool

- FG - “Filling-the-Gap”
  - Batch offline analysis, support for Condor
  - Open Source Software
  - MCR executables (BSD-3)
- Main repo:
  - <https://github.com/Imperial-AESOP/Filling-the-Gap>
- Manual available in the repo

## ■ Outputs

- Model parameters
- Visualization
- Forecasting
  - Requires analysis, but not decision-making

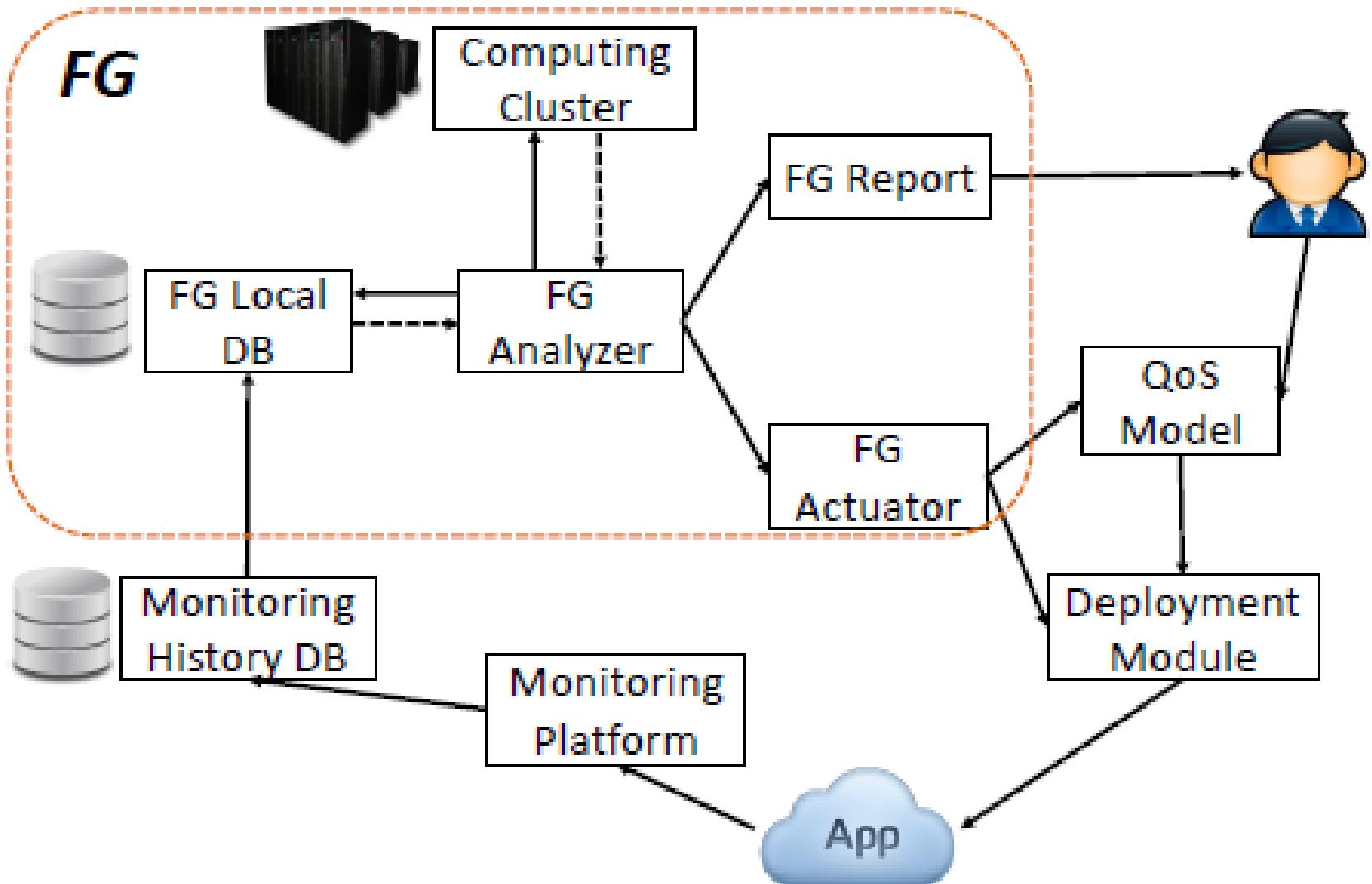
## ■ User control knobs

- Analysis frequency
- Horizon of analysis
- Monitoring intensity
- Maximum collection window

## ■ Parameters for QN/LQN models

Parameter	Data Required	Level		Platform	
		App.	VM	IaaS	PaaS
Population	Total Number of requests	X		X	X
Resource Consumption	Utilization		X	X	
	Throughput	X		X	X
	Queue length	X		X	X
	Response Times	X		X	X
	Queue length (arrival)	X		X	X
Think time	Throughput	X		X	X
	Total Number of requests	X		X	X
	Mean Number Requests	X		X	X
Stage duration, transition probs. and efficiency	Start-up duration		X	X	
	Availability (Up/Down)		X	X	
	CPU Steal		X	X	

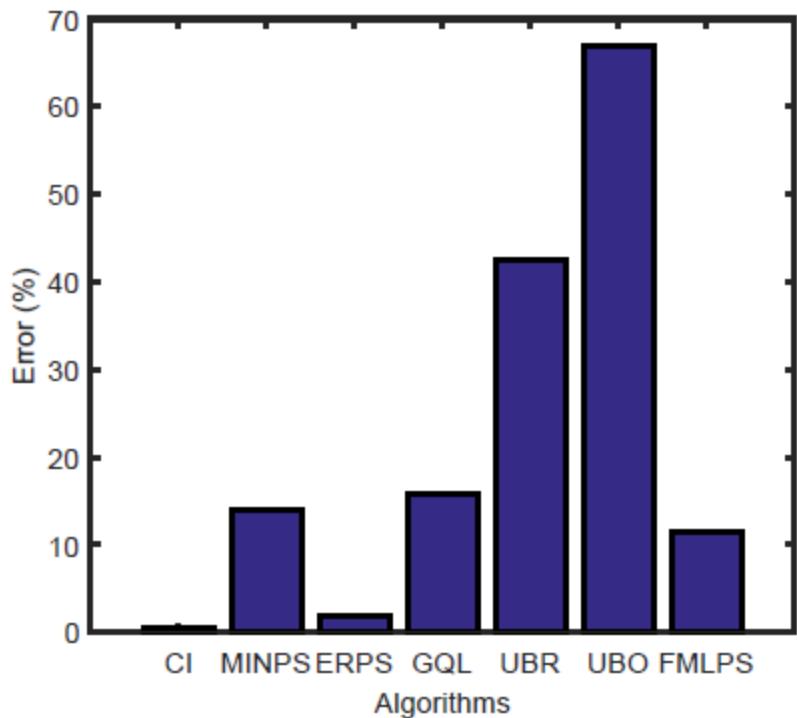
# FG: Architecture



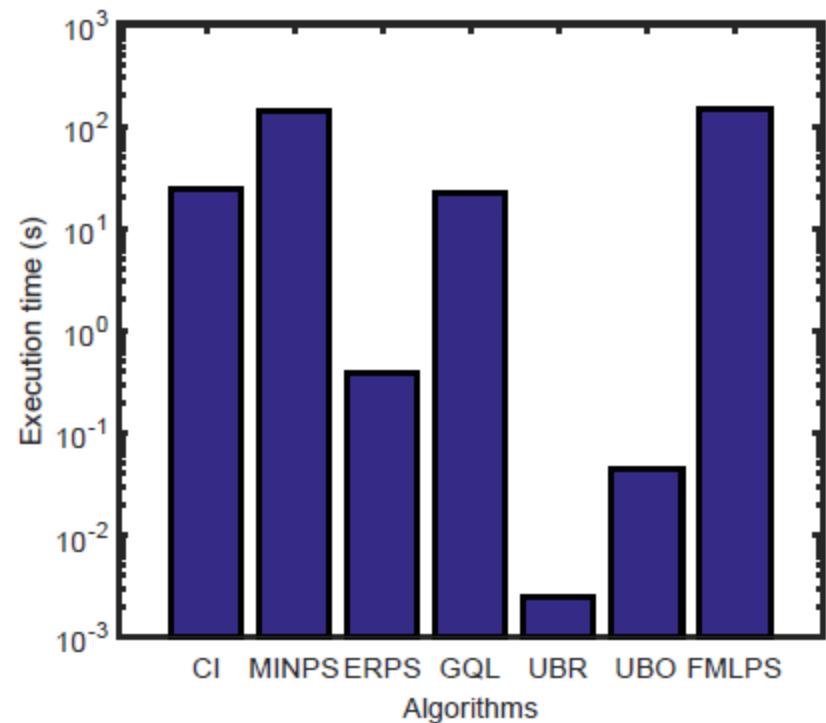
## ■ Implemented methods

- Complete Information (**CI**)
- Utilization-Based Regression (**UBR**)
- Utilization-Based Optimization (**UBO**)  
(M/GI/1-PS; cf. Zhang et al., Menasce)
- ODE-based MLPS (“fluid MLPS”, **FMLPS**)
- **MINPS**
- Queue-Based Gibbs Sampling (**GQL**)
- Extended RPS (**ERPS**, includes a new correction for number of cores)

# FG: Comparison Results



(a) Error (%)



(b) Execution time (s)

# Queue-Length Based Estimation

# Gibbs Sampling (GQL)

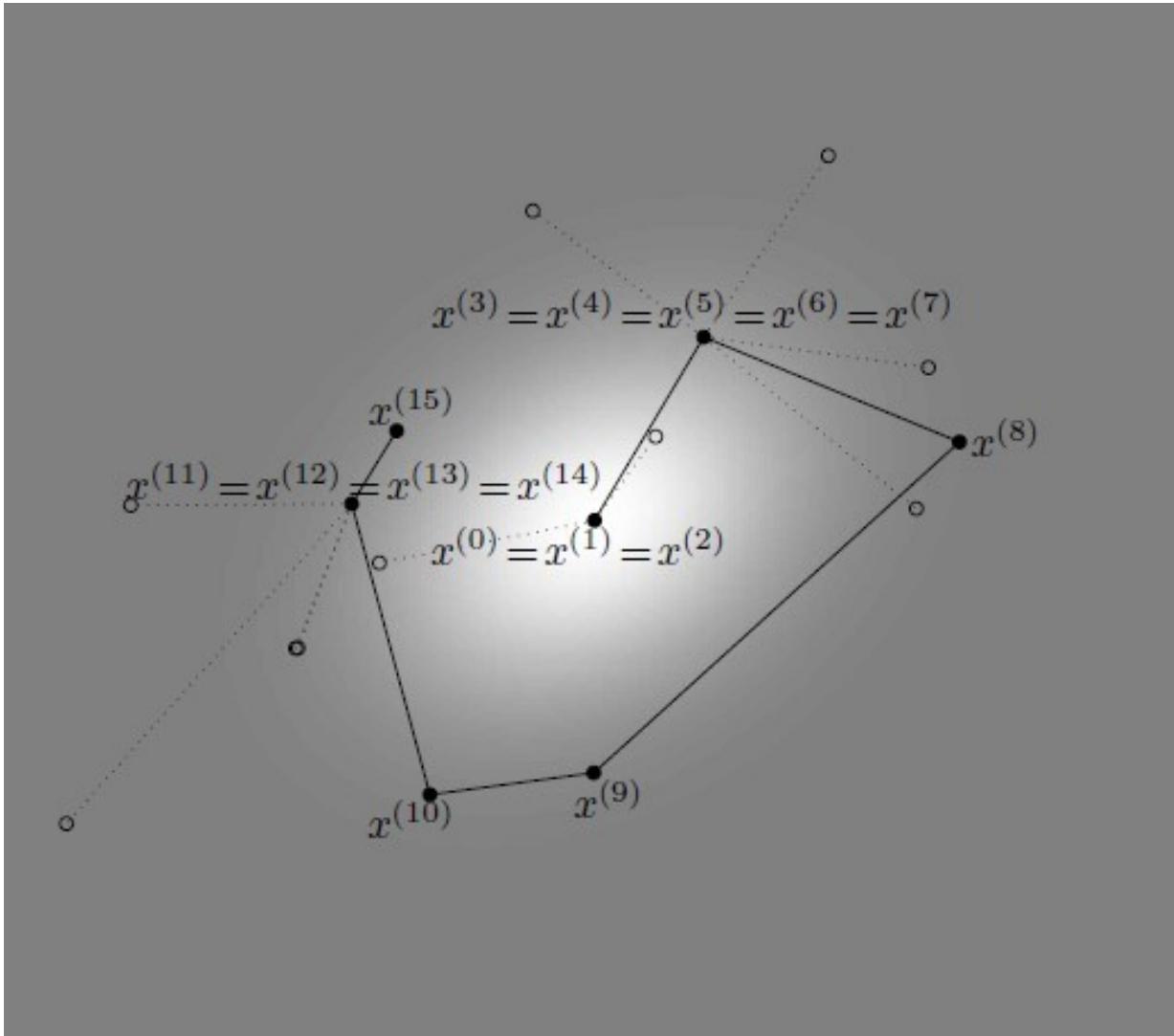
- Monitor occupancy at all resources
  - Observations:  $\{\mathbf{n}^{(1)}, \mathbf{n}^{(2)}, \dots, \mathbf{n}^{(L)}\}$
  - Ill-posed, unless think times known
- Probabilistic model of distributed system

$$P(\mathbf{n}|\mathbf{D}) = \frac{Z_1^{n_{01}}}{n_{01}!} \cdots \frac{Z_R^{n_{0R}}}{n_{0R}!} \prod_{i=1}^V n_i! \prod_{r=1}^R \frac{D_{ir}^{n_{ir}}}{n_{ir}! G(\mathbf{D})}$$

- Markov-Chain Monte-Carlo (MCMC)
  - draw samples from target distribution  $P(\mathbf{D}|\mathbf{n})$
  - averaging samples provides estimate  $\bar{\mathbf{D}}$

# Gibbs Sampling (GQL)

## ■ Markov-Chain Monte-Carlo (MCMC)



# Gibbs Sampling (GQL)

- Gibbs: sample one dimension at a time

- $P(D_{ir}|D_{01}^s, \dots, D_{i,r-1}^s, D_{i,r+1}^{(s-1)}, \dots, D_{VR}^{(s-1)}, \mathbf{N}) \rightarrow D_{ir}^{(s)}$

- iteratively sample posterior

**prior**

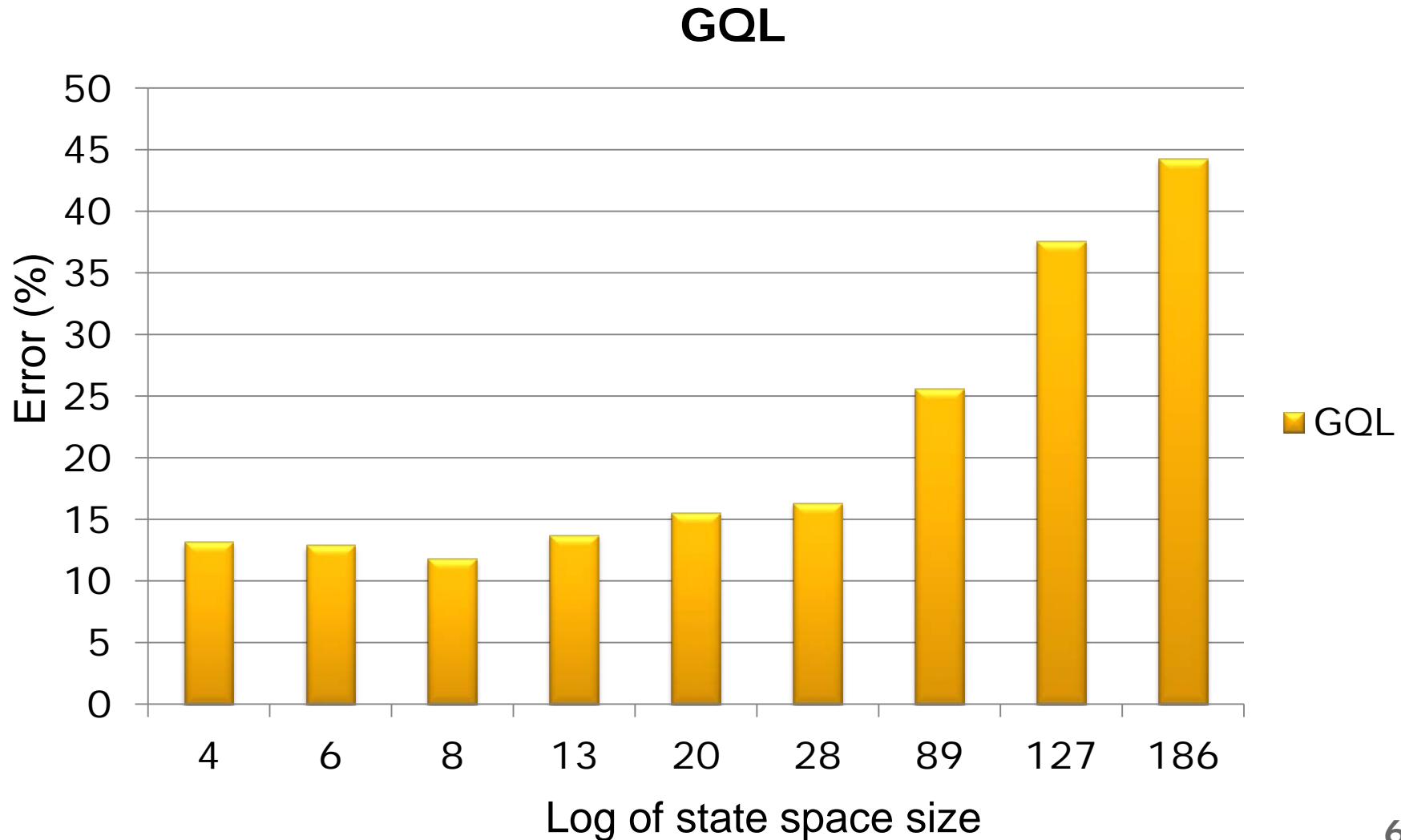
$$\prod_{i=1}^L P(D_{ir}|\mathbf{D}_{(ir)}, \mathbf{n}^{(i)}) = \prod_{i=1}^L \frac{P(\mathbf{n}^{(i)}|\mathbf{D})P(\mathbf{D})}{\int_{D_{ir}} P(\mathbf{n}^{(i)}|\mathbf{D})P(\mathbf{D})dD_{ir}}$$

- where  $\mathbf{D}_{(ir)} = \mathbf{D} \setminus D_{ir}$

- Accurate estimates, error ~3%-7%

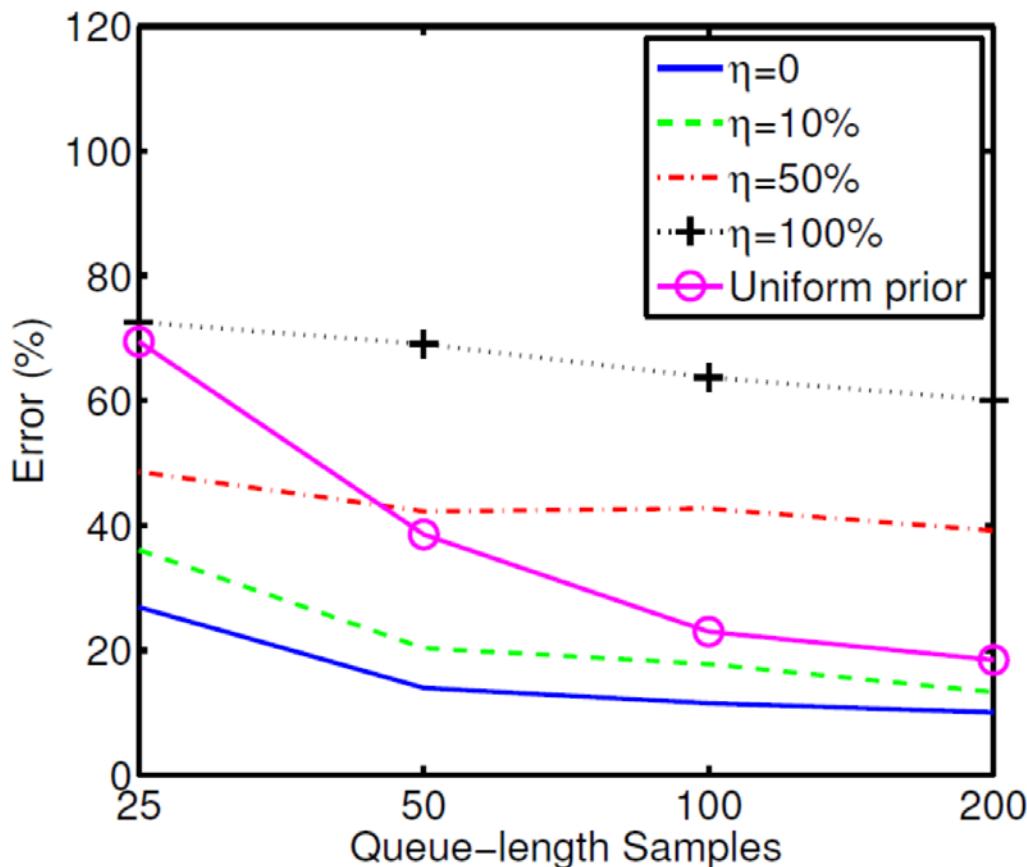
$R$	$V$	Time(s)	Average	C.I.
1	1	38	4.27%	4.91% - 3.64%
1	3	148	3.45%	3.75% - 3.16%
2	1	83	3.77%	4.17% - 3.37%
4	4(Ser)	1489	5.05%	5.22% - 4.88%
4	4(Pal)	1460	7.48%	7.16% - 7.79%

## ■ Increasing model sizes

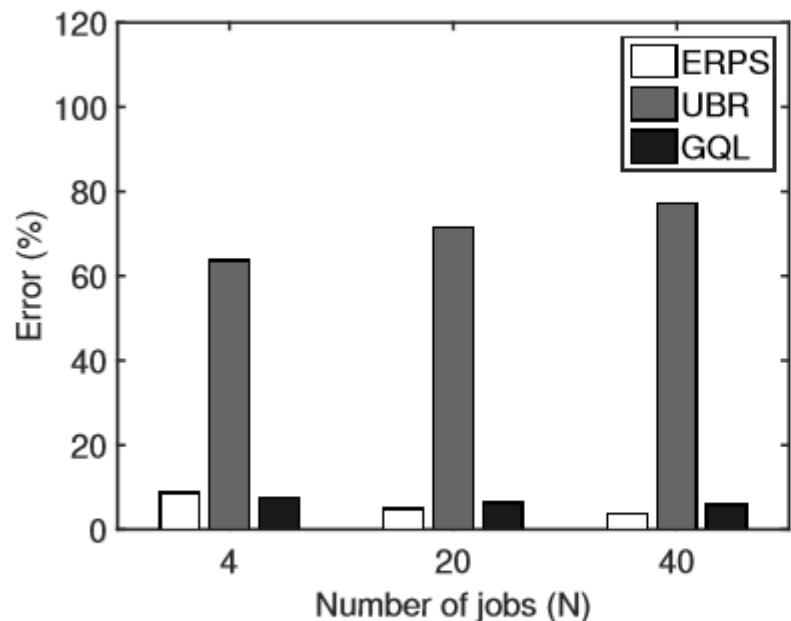


## ■ Dirichlet prior

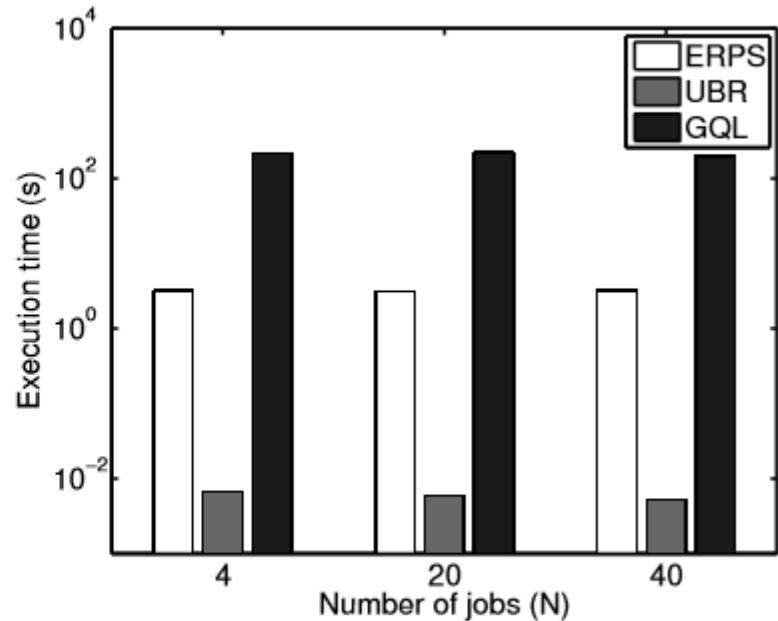
- all estimates converge unless the exact demand has very low probability prior



■ Accurate estimates, error ~3%-7%



(a) Estimation Error



(b) Execution time

# GQL vs Other MCMC Methods

- Far better convergence properties than Metropolis-Hastings and Slice sampling

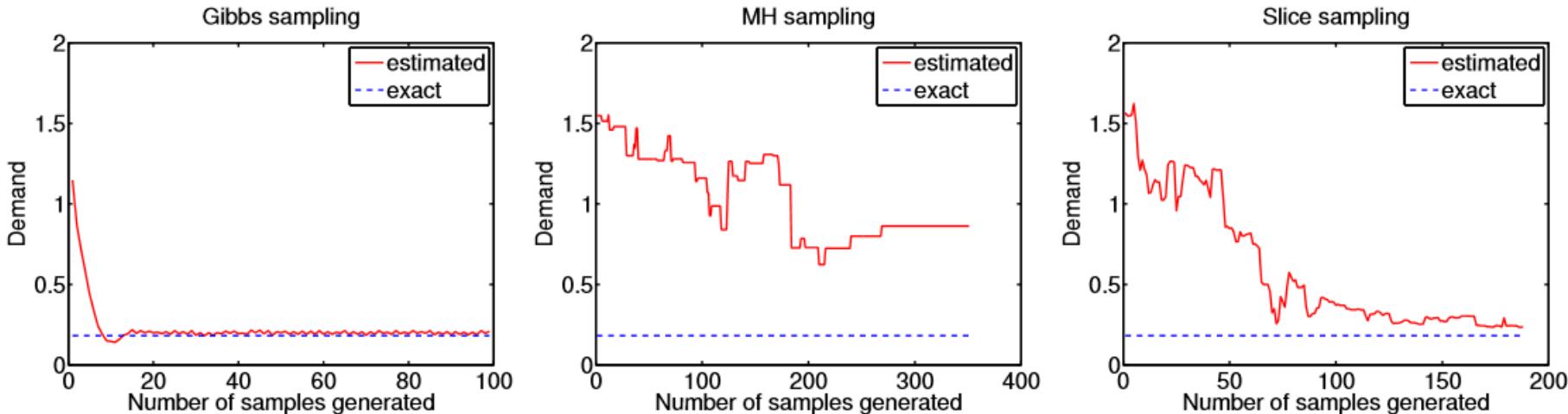


Fig. 4: Convergence plots

- About 13-15% error in estimating demands against cloud ERP data (Apache OFBiz)

- Based on Maximum Likelihood Estimation
- Works with mean queue length
- A simple approximation of the MLE:

- Consider the demand vector  $\theta^{bs}$  where

$$\theta_{ij}^{bs} = \frac{\tilde{Q}_{ij}(D)}{(N_j - \sum_{k=1}^M \tilde{Q}_{kj})} \frac{\theta_{0,j}}{(1 + \sum_{h=1}^R \tilde{Q}_{ih} - \tilde{Q}_{ij}(D)/N_j)}$$

- Approach generalizes to load-dependent QNs
- More details at tomorrow's talk!

# FG Tool

- FG - “Filling-the-Gap”
  - Batch offline analysis, support for Condor
  - Open Source Software
  - MCR executables (BSD-3)
- Main repo:
  - <https://github.com/Imperial-AESOP/Filling-the-Gap>
- Manual available in the repo

## ■ Outputs

- Model parameters
- Visualization

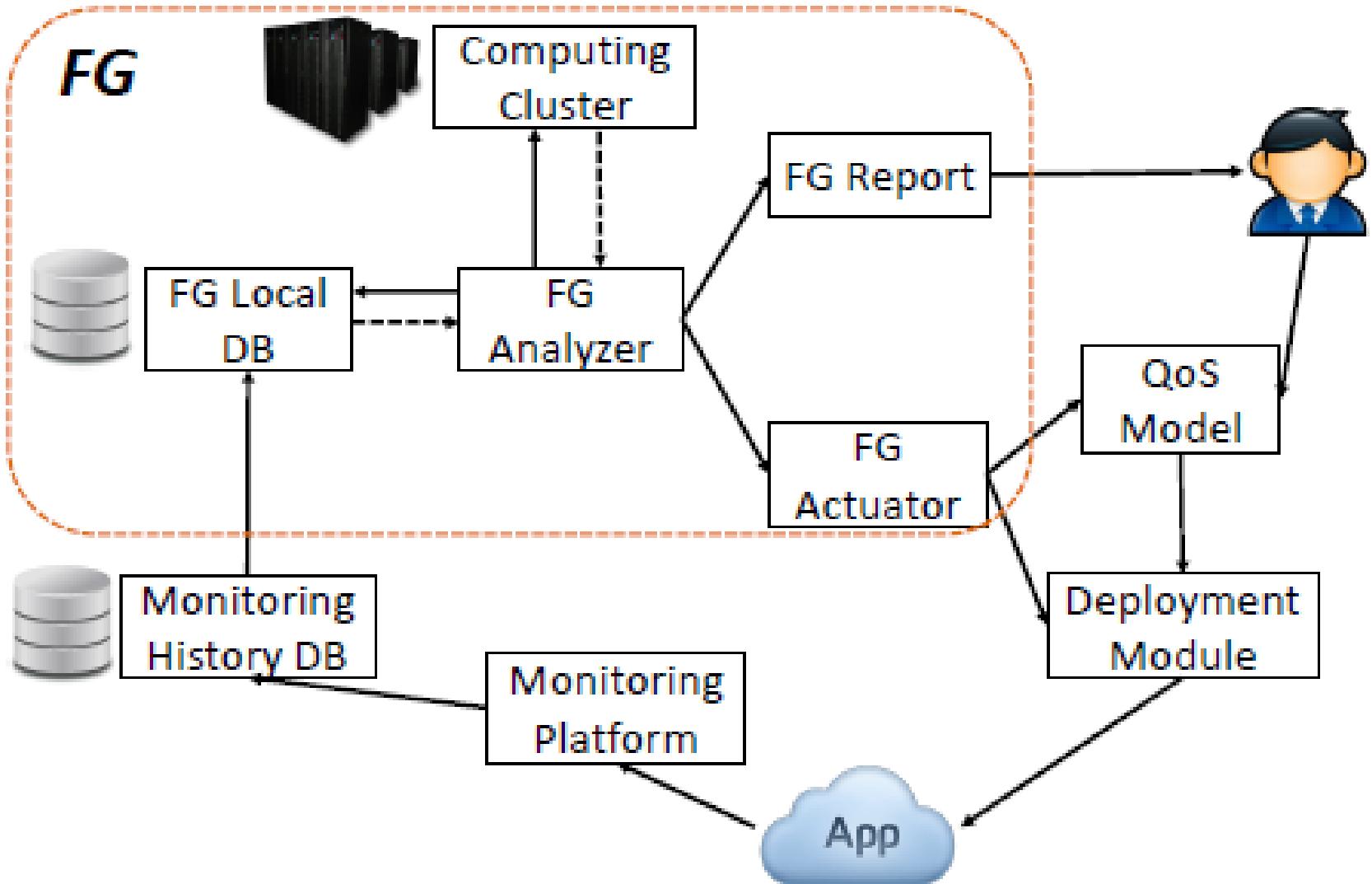
## ■ User control knobs

- Analysis frequency
- Horizon of analysis
- Algorithm selection

## ■ Parameters for QN/LQN models

Parameter	Data Required	Level		Platform	
		App.	VM	IaaS	PaaS
Population	Total Number of requests	X		X	X
Resource Consumption	Utilization		X	X	
	Throughput	X		X	X
	Queue length	X		X	X
	Response Times	X		X	X
	Queue length (arrival)	X		X	X
Think time	Throughput	X		X	X
	Total Number of requests	X		X	X
	Mean Number Requests	X		X	X

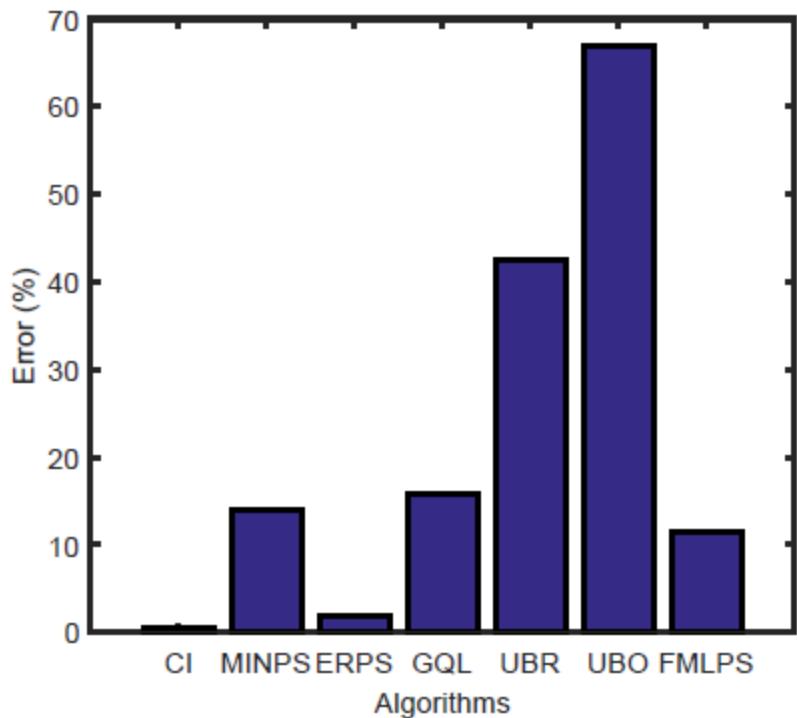
# FG: Architecture



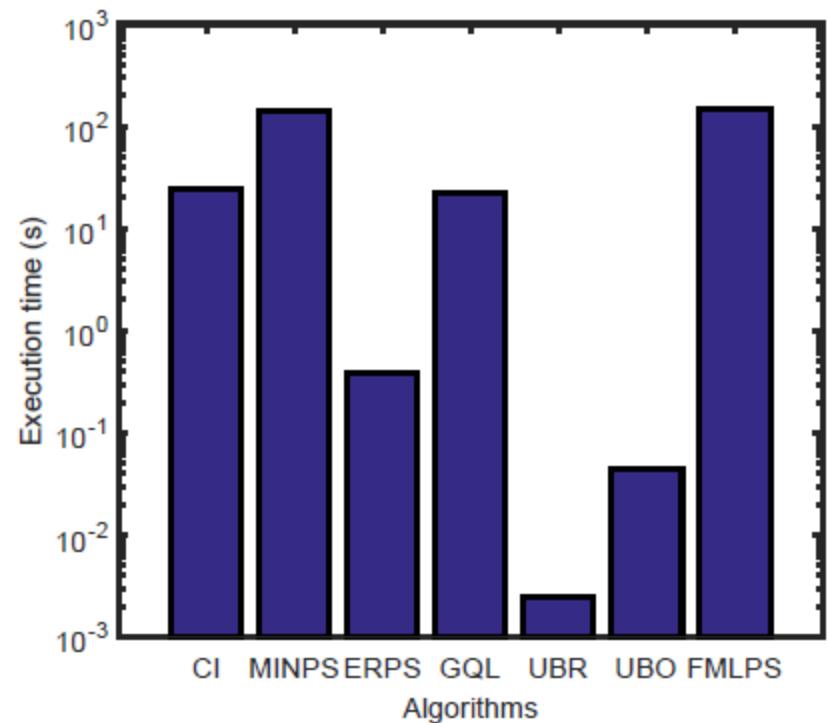
## ■ Implemented methods

- Complete Information (**CI**)
- Utilization-Based Regression (**UBR**)
- Utilization-Based Optimization (**UBO**)  
(M/GI/1-PS; cf. Zhang et al., Menasce)
- ODE-based MLPS (“fluid MLPS”, **FMLPS**)
- **MINPS**
- Queue-Based Gibbs Sampling (**GQL**)
- Extended RPS (**ERPS**, includes a new correction for number of cores)

# FG: Comparison Results



(a) Error (%)



(b) Execution time (s)

# Comparison & Case Studies

## Elsevier PEVA, October 2015.

### Evaluating Approaches to Resource Demand Estimation

Simon Spinner<sup>a,\*</sup>, Giuliano Casale<sup>b</sup>, Fabian Brosig<sup>a</sup>, Samuel Kounnev<sup>a</sup>

<sup>a</sup>*University of Würzburg, Am Hubland, Würzburg, Germany*

<sup>b</sup>*Imperial College London, Department of Computing, SW7 2AZ, UK*

---

#### Abstract

Resource demands are a key parameter of stochastic performance models that needs to be determined when performing a quantitative performance analysis of a system. However, the direct measurement of resource demands is not

# Experiments

## ■ Dataset D1: Queueing Simulator

- Simulated M/M/1 queue with FCFS scheduling
- Workload classes: 1, 2 and 5
- Utilization levels: 10%, 50%, 90%
- In total: 900 traces

## ■ Dataset D2: Micro-Benchmarks

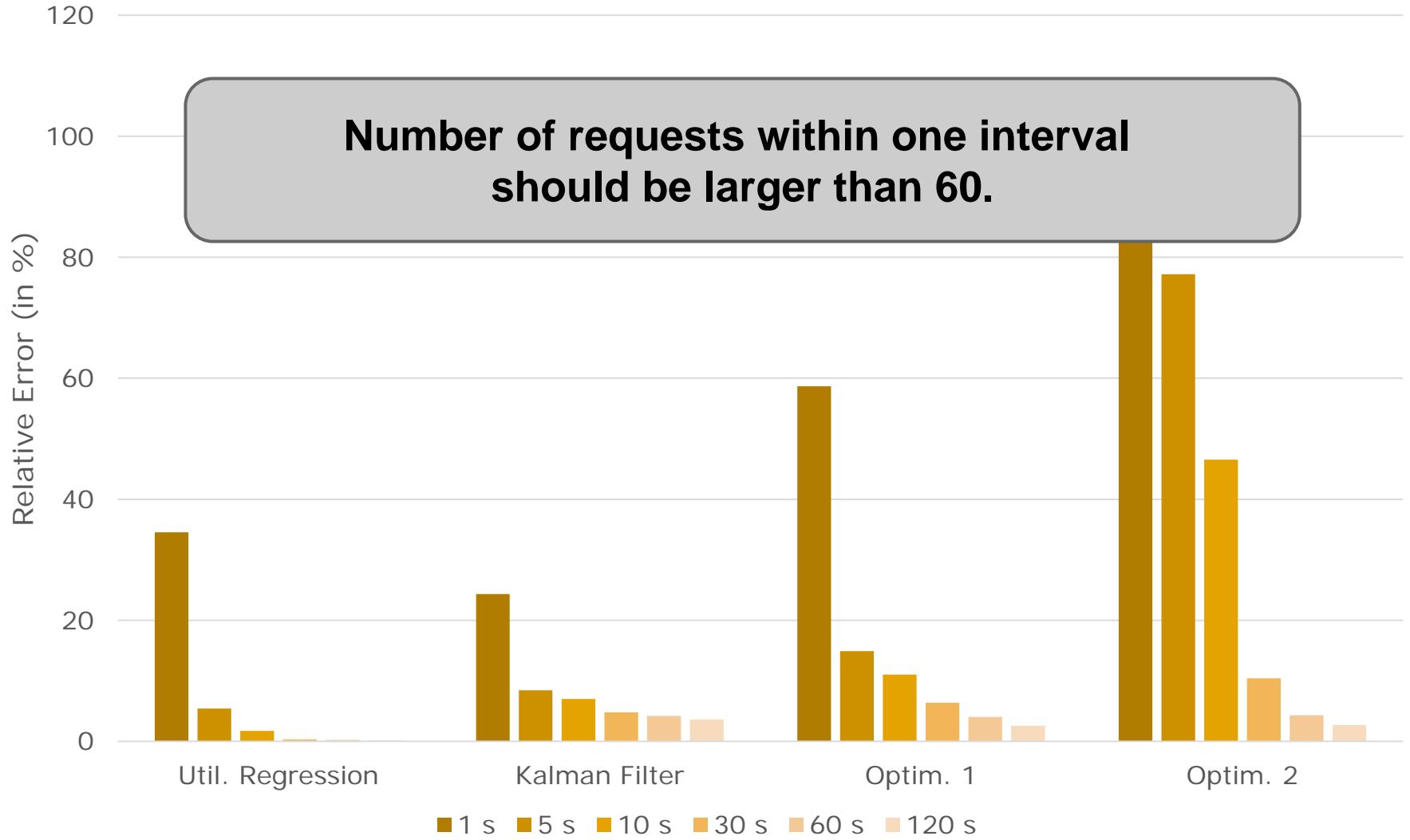
- Workload classes: 1, 2, and 3
- Utilization levels: 20%, 50%, 80%
- In total: 210 traces

# Compared Approaches

- Based on Service Demand Law (Brosig et al. 2009)
- Utilization Regression (Rolia and Vetland 1995)
- Kalman Filter (Kumar et al. 2009)
- Optimization 1 (Menascé 2008)
- Optimization 2 (Liu et al. 2006)
- Response time regression (Kraft et al. 2009)
- Gibbs Sampling (Wang et al. 20133)

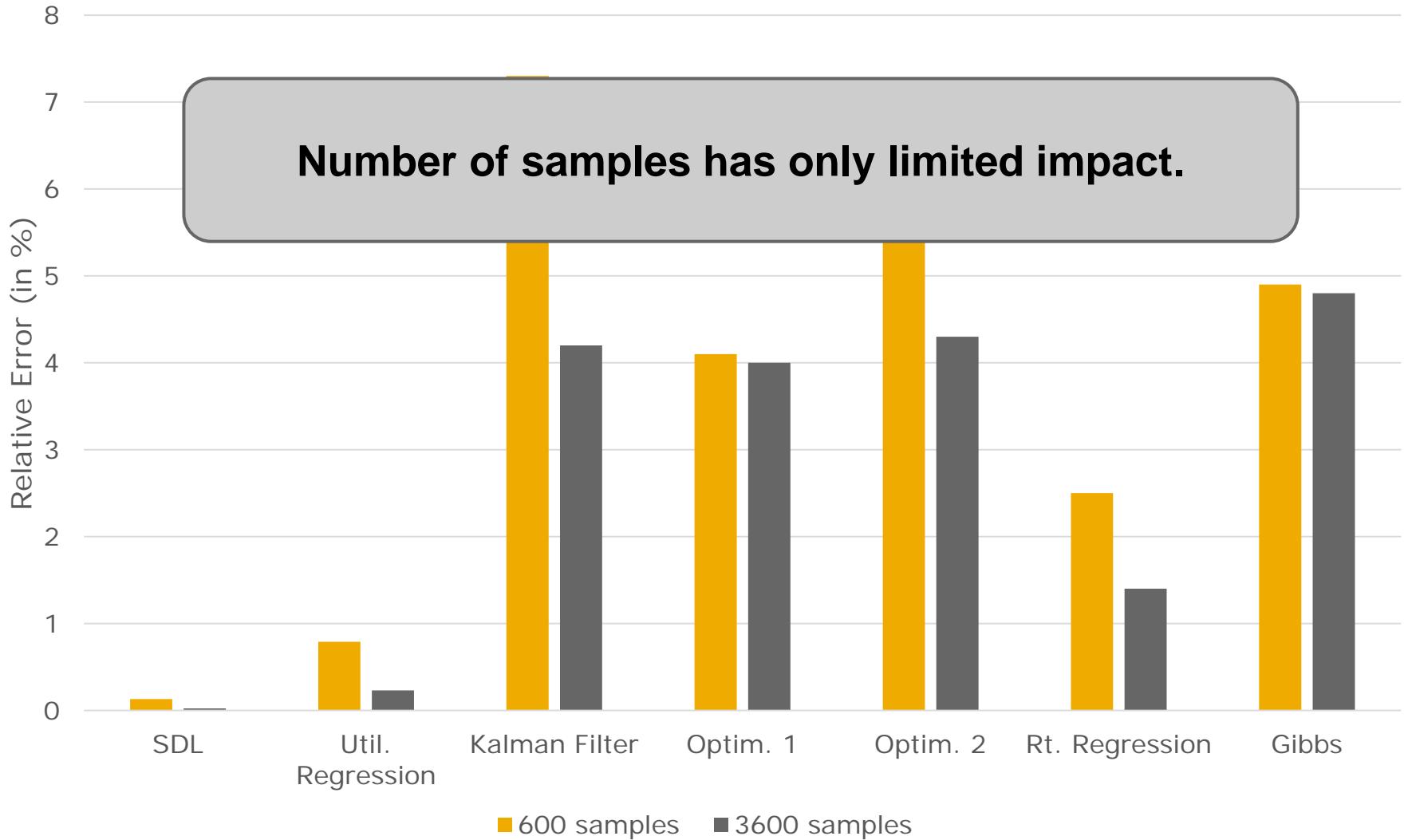
# Sampling Interval

Dataset D1



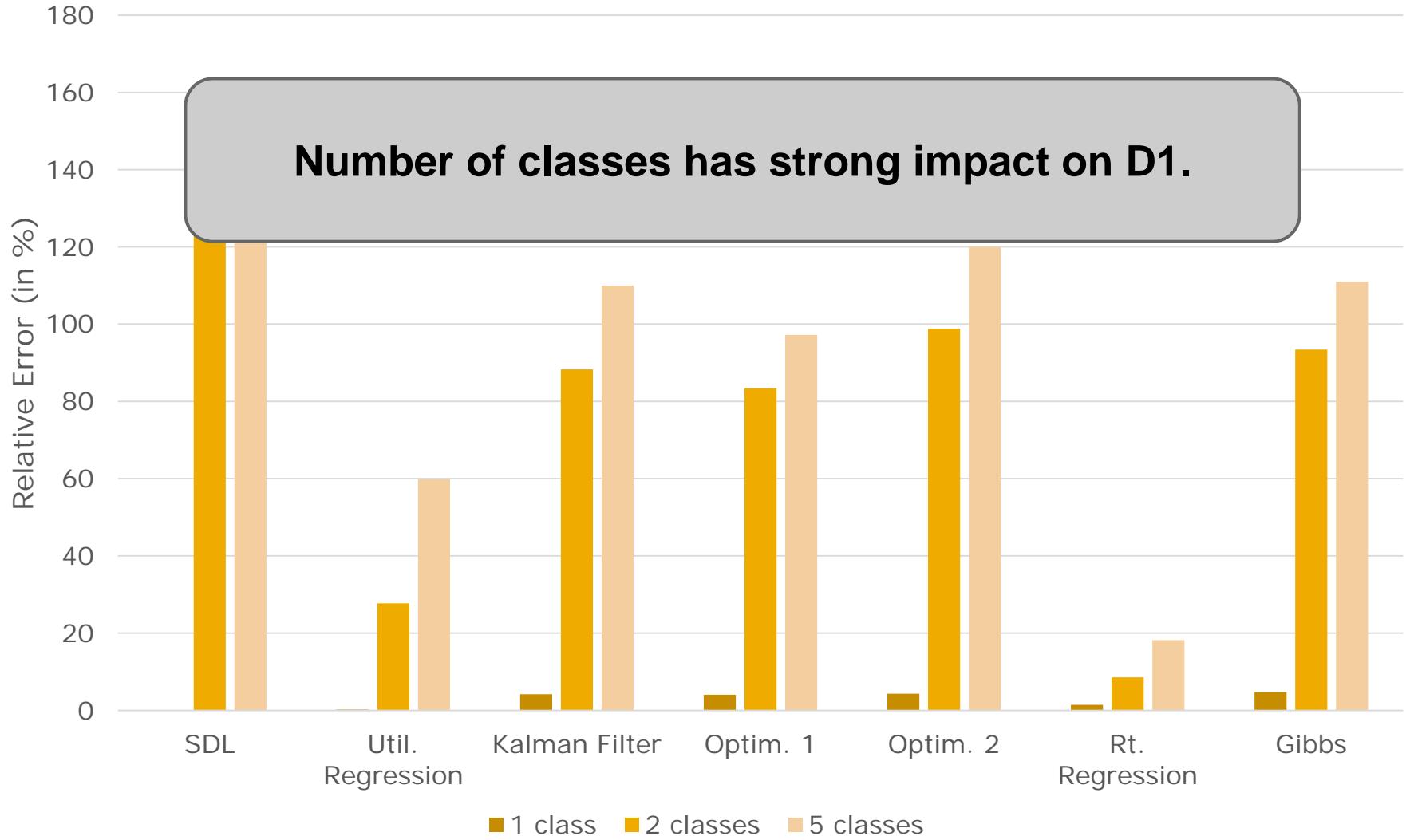
# Number of Samples

Dataset D1



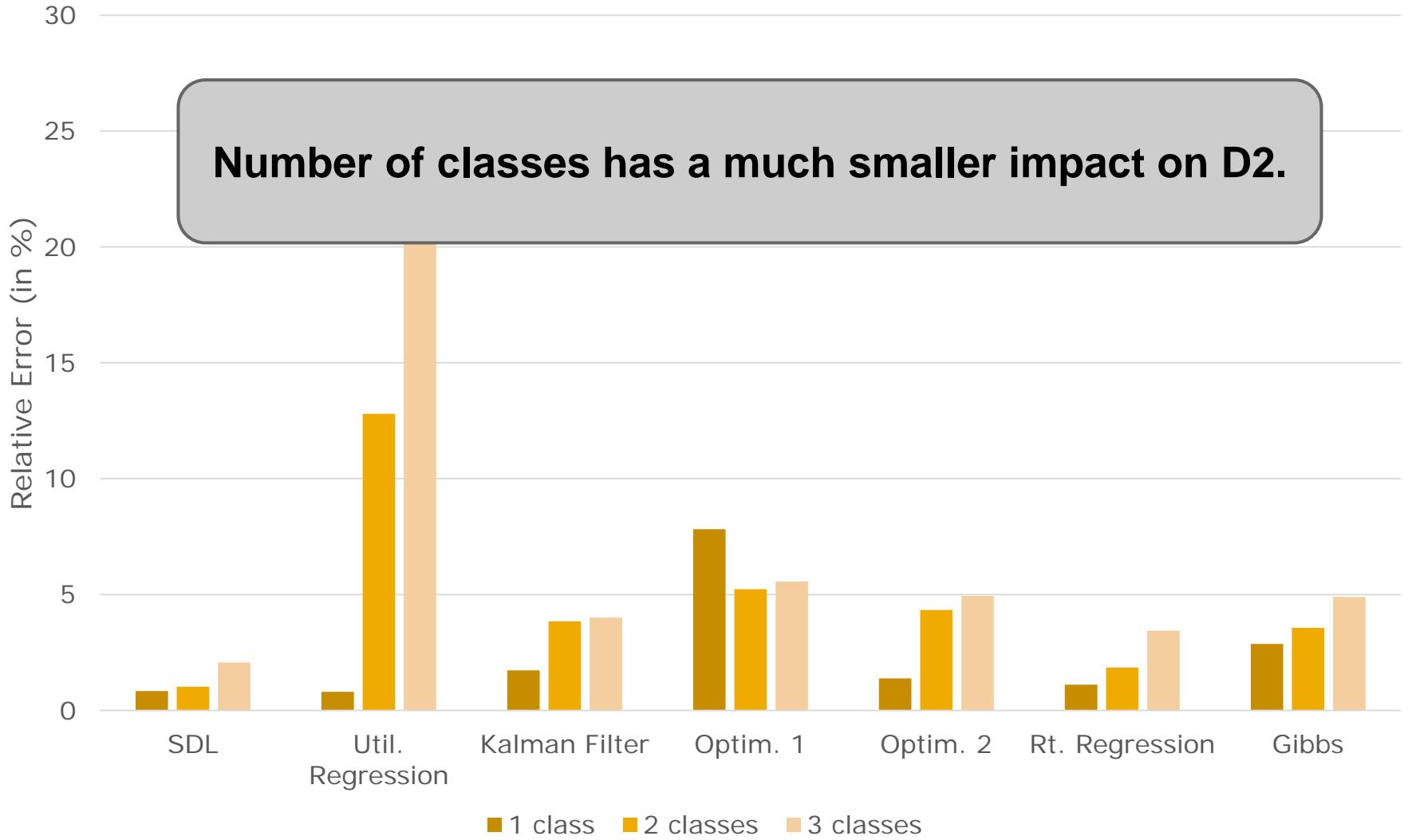
# Number of Workload Classes

Dataset D1



# Number of workload classes

Dataset D2



# Correlation Analysis

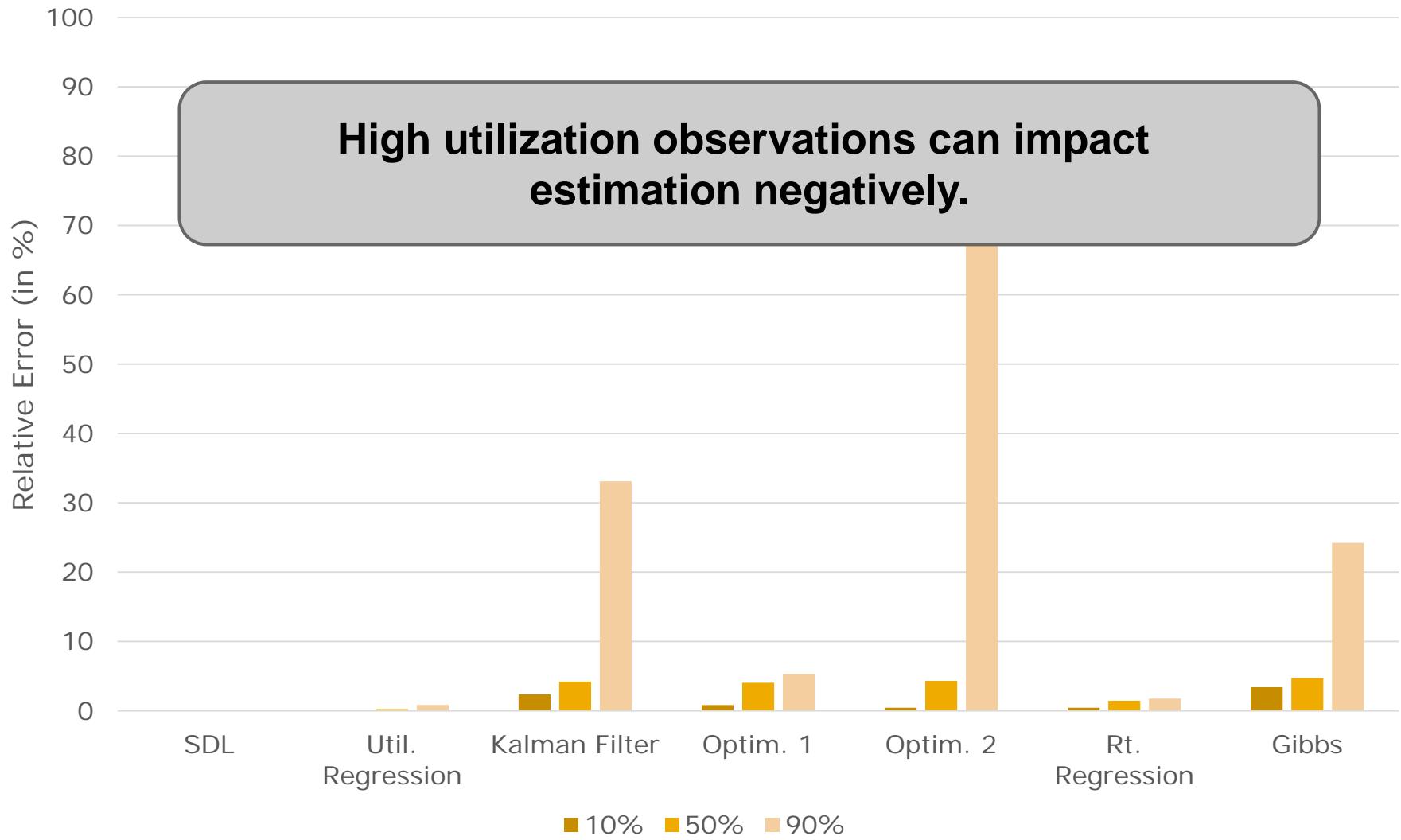
## Correlation with std[D]:

Classes	SDL	Util. Regression	Kalman Filter	Optim. 1	Optim. 2	Rt. Regression	Gibbs
2	0.91	0.35	0.88	0.89	0.88	0.52	0.9
5	0.72	0.37	0.78	0.8	0.8	0.44	0.79

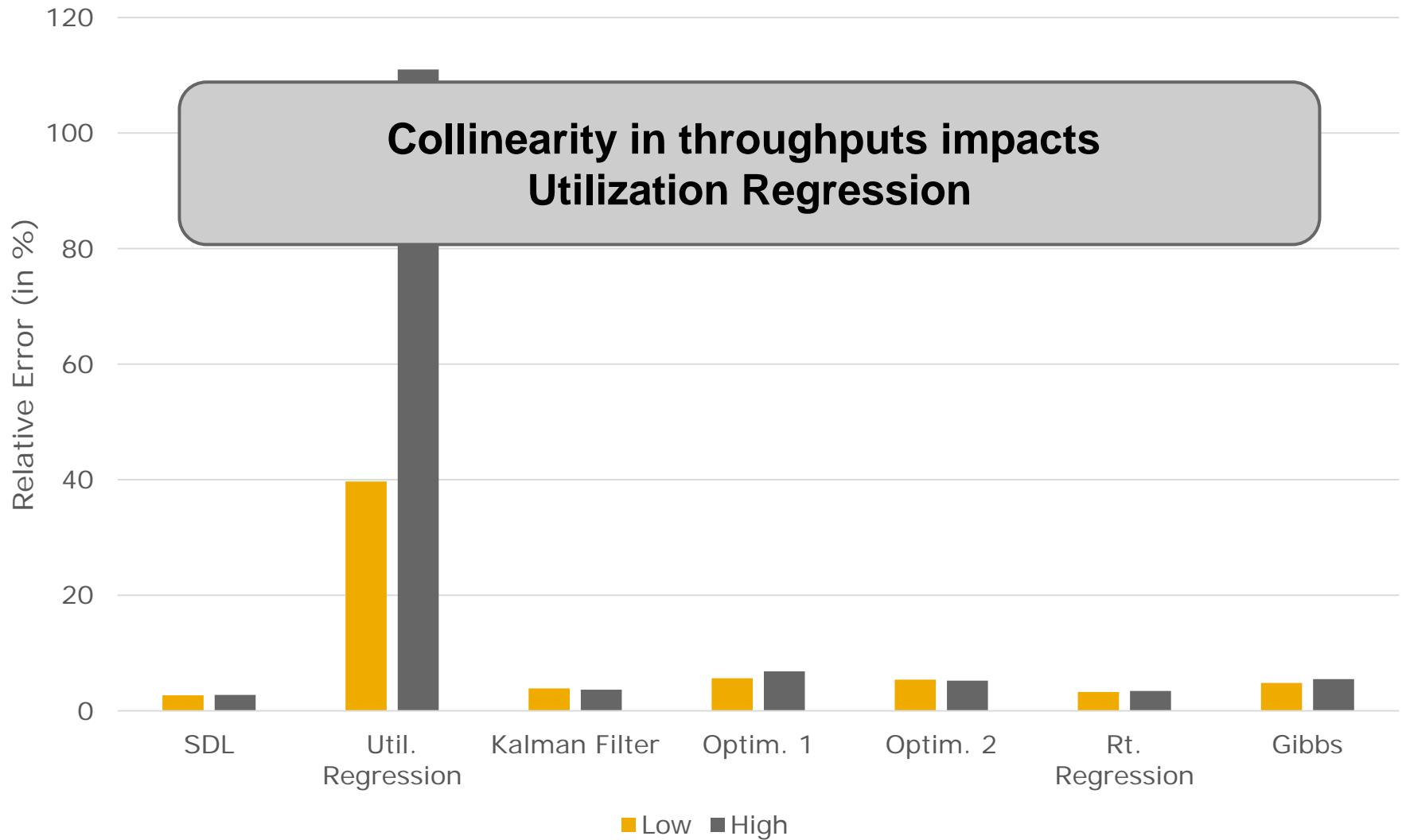
Significant influence of scheduling strategy.

# Load Level

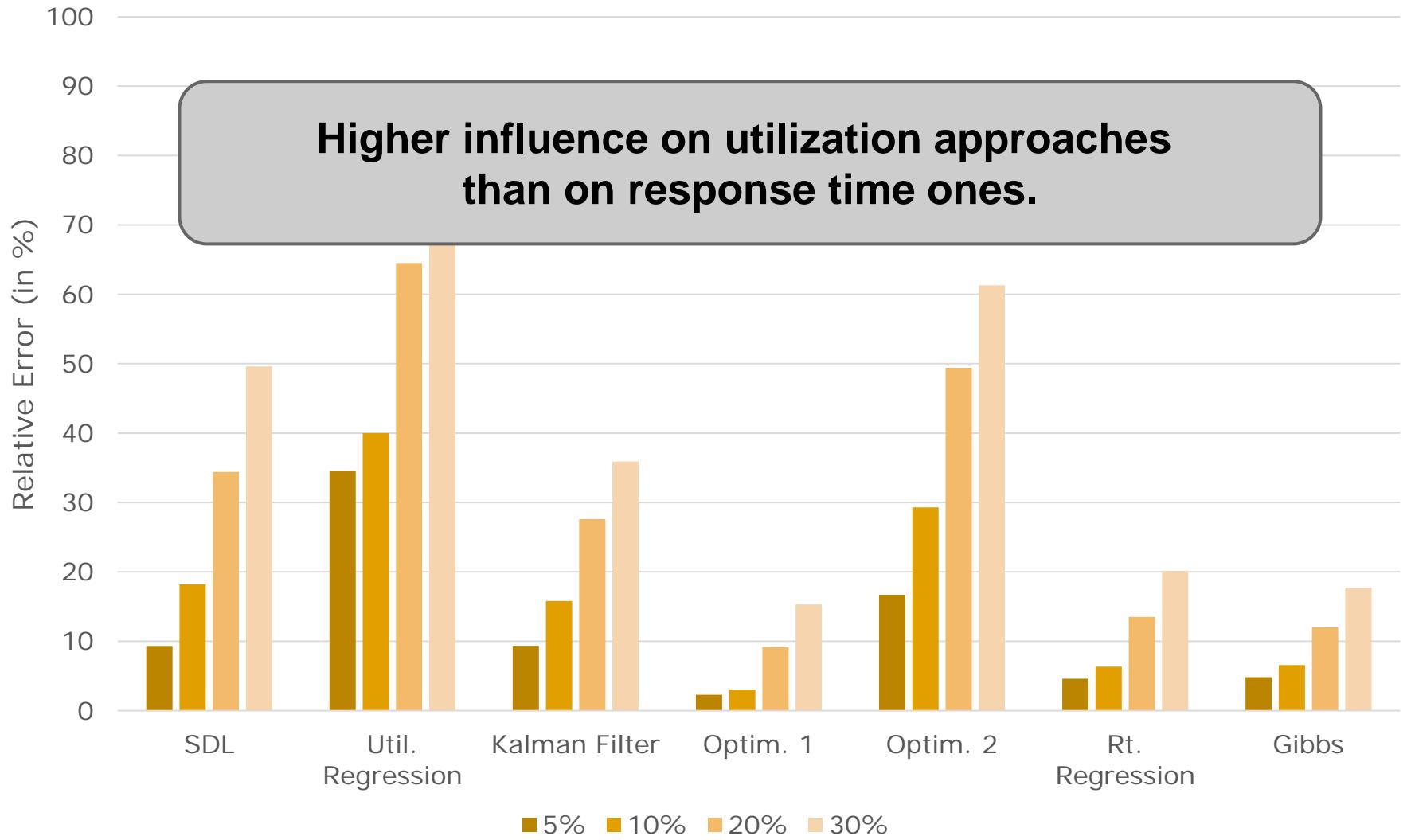
Dataset D1



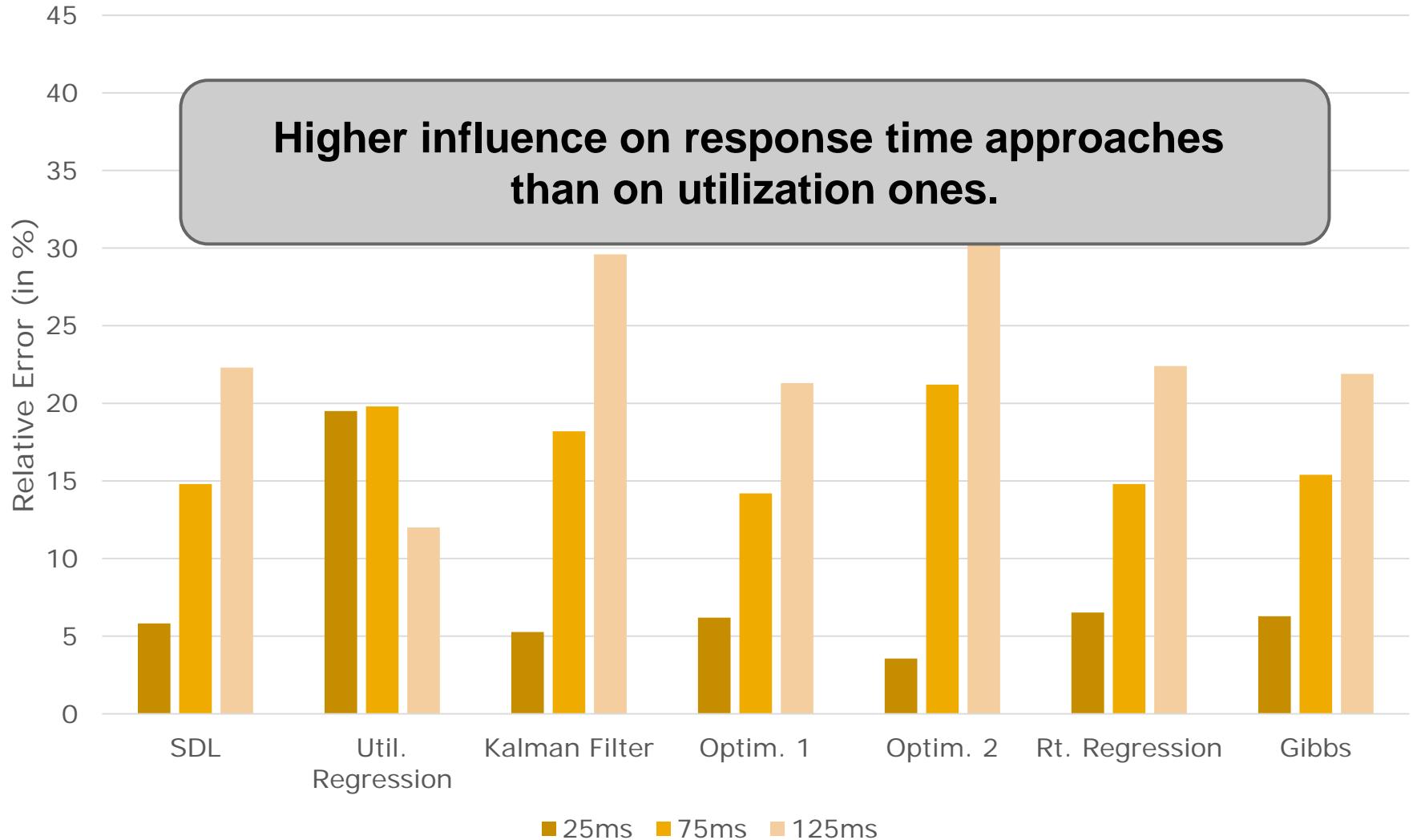
# Collinearity



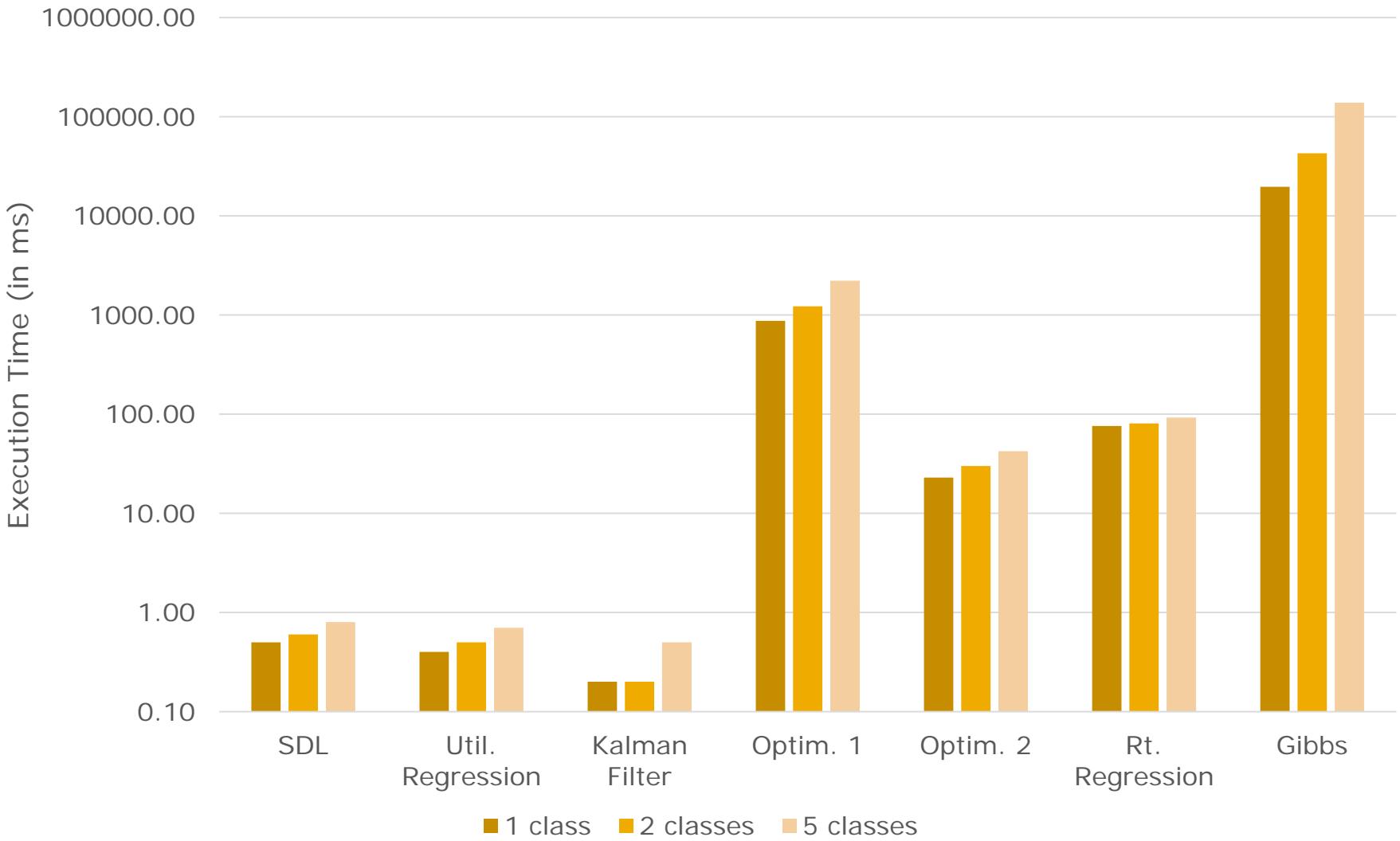
# Missing Jobs



# Additional Wait Times

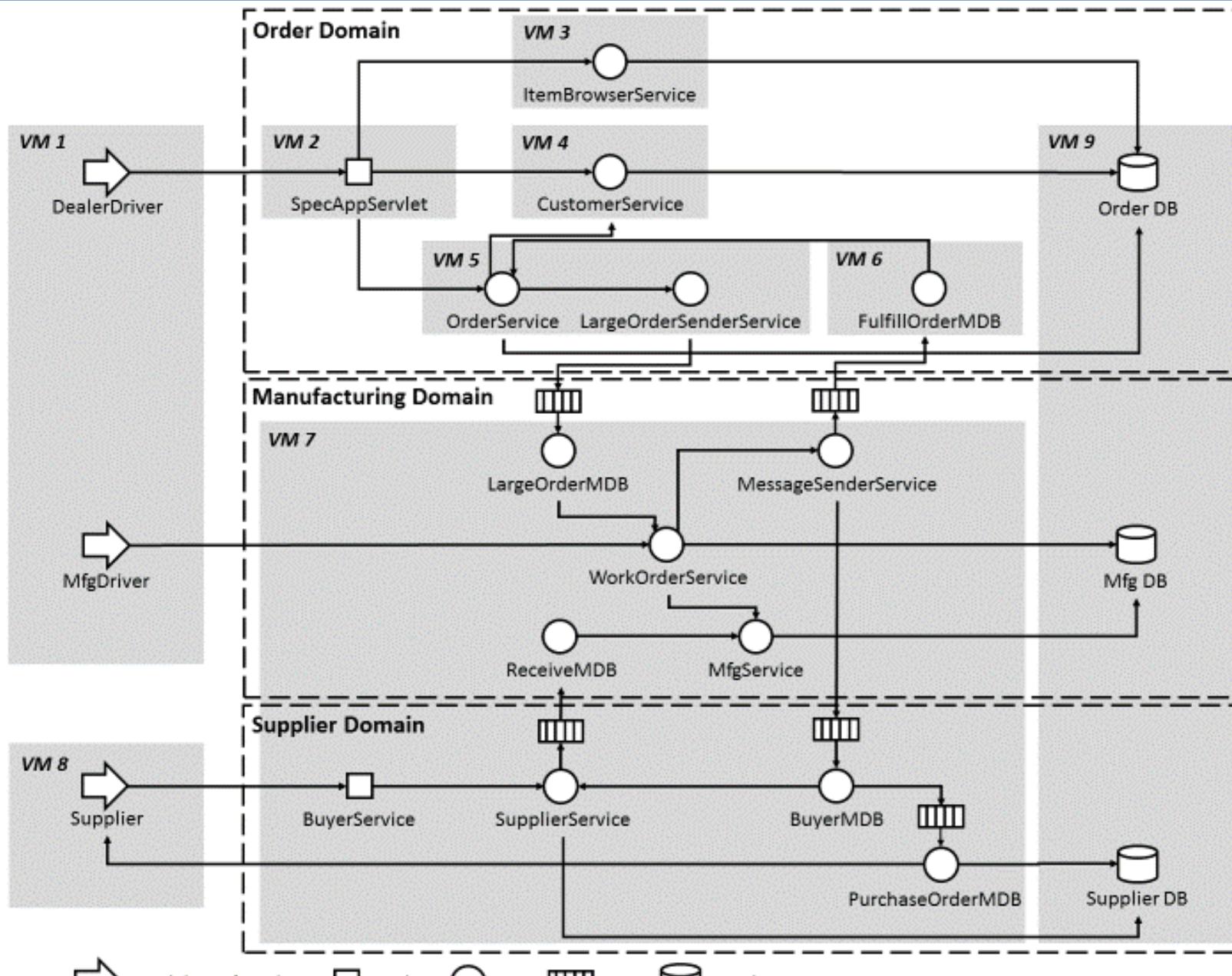


# Execution Time

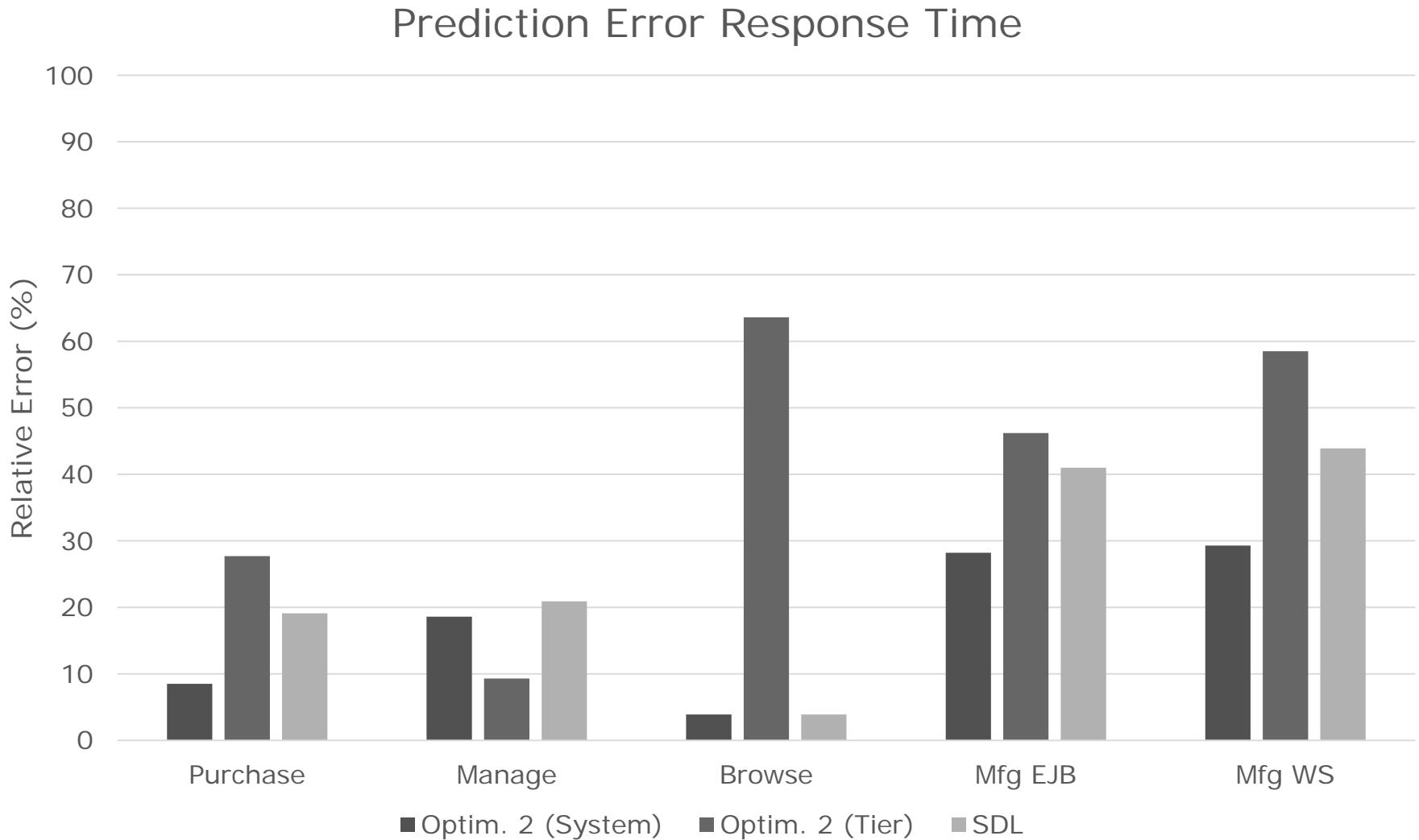


- SPECjEnterprise2010 application benchmark
  - Distributed deployment over 7 VMs
  - „Microservice Style“
- Strategies for Demand Estimation
  - Observed end-to-end response time
  - Observed residence time per tier
  - Per-resource statistics

# Experiment Setup

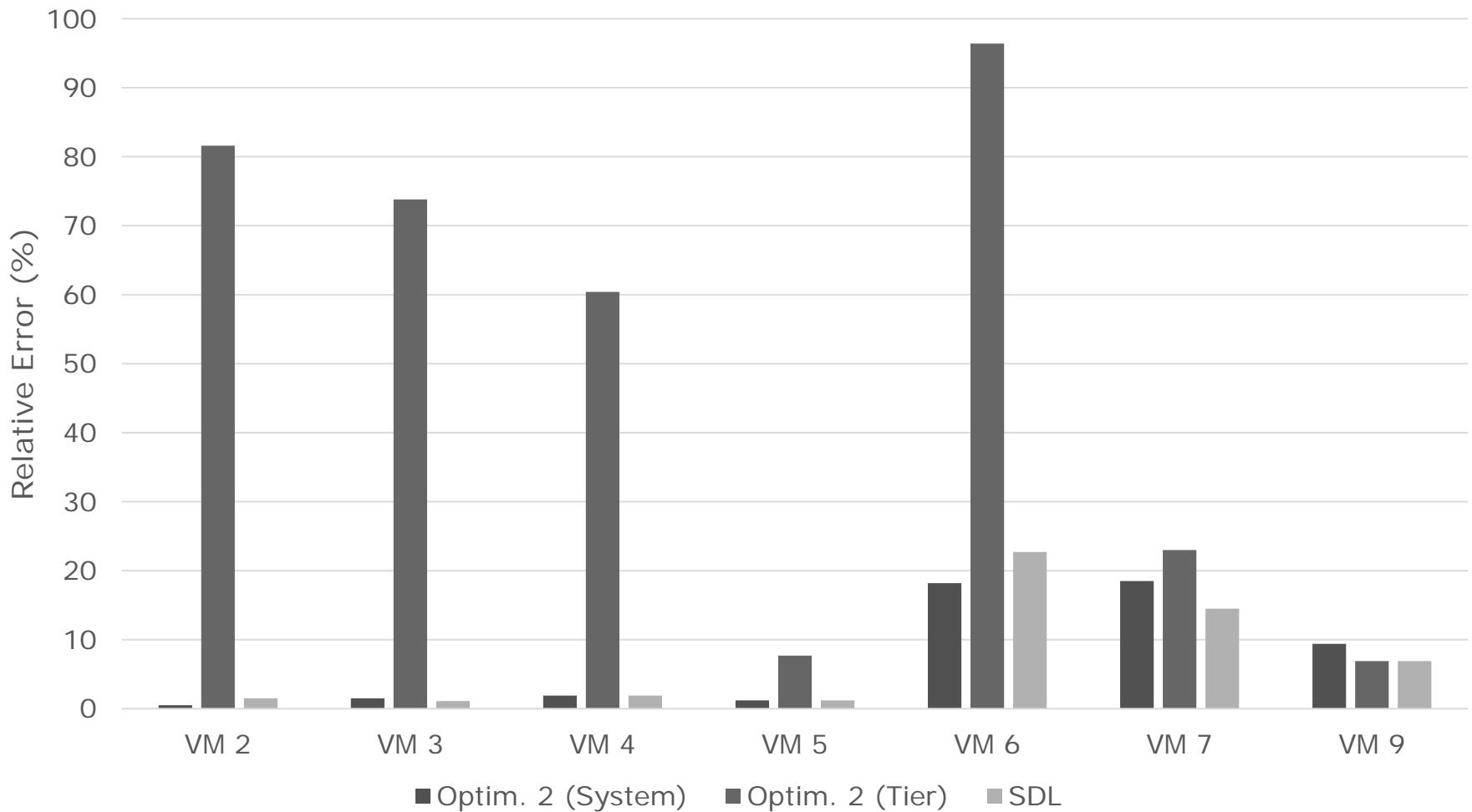


# Results



# Results

## Prediction Error Utilization



## ■ Admission control

- Multi-tenant application (extended TPC-W)
- SAP HANA cloud platform

## ■ Supports Performance isolation between tenants

IEEE/ACM CCGrid 2014.

### Resource Usage Control In Multi-Tenant Applications

Rouven Krebs

SAP AG

HANA - Applied Research  
69190 Walldorf  
Germany  
rouven.krebs@sap.com

Simon Spinner

Karlsruhe Institute of Technology  
IPD  
76131 Karlsruhe  
Germany  
simon.spinner@kit.edu

Nadia Ahmed

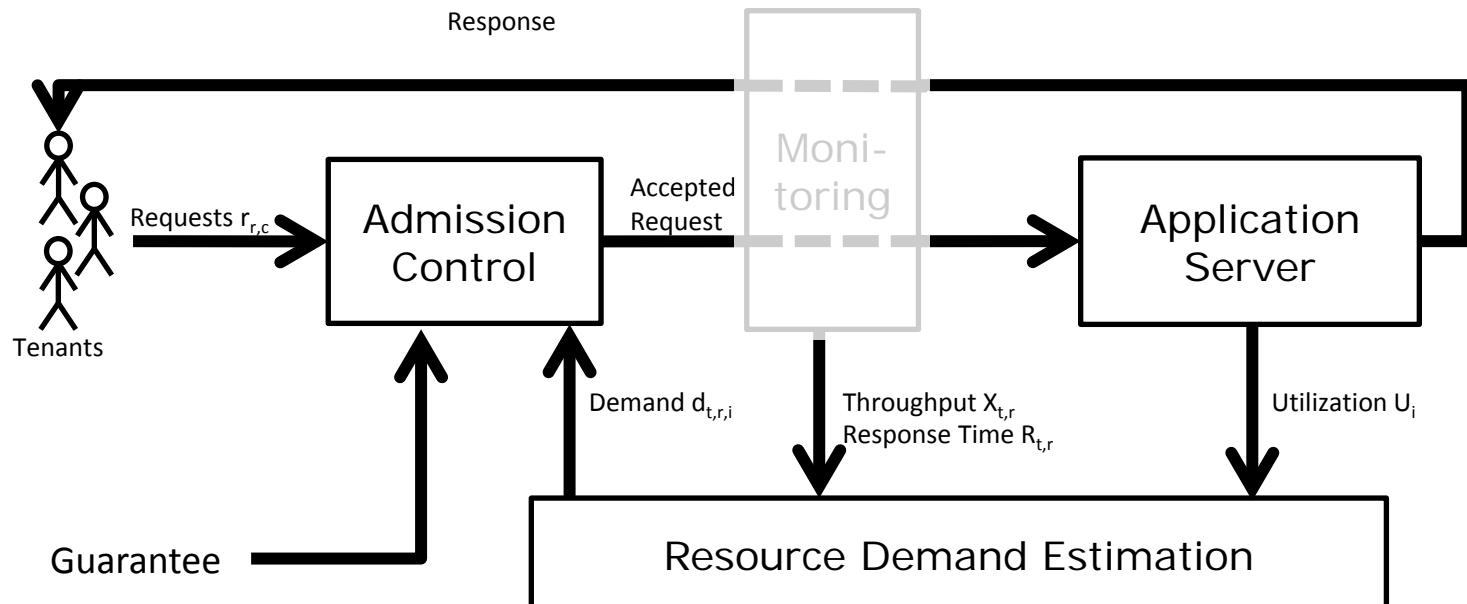
SAP AG

HANA - Applied Research  
69190 Walldorf  
Germany  
nadia.ahmed@sap.com

Samuel Kounev

Karlsruhe Institute of Technology  
IPD  
76131 Karlsruhe  
Germany  
kounev@kit.edu

# General Idea



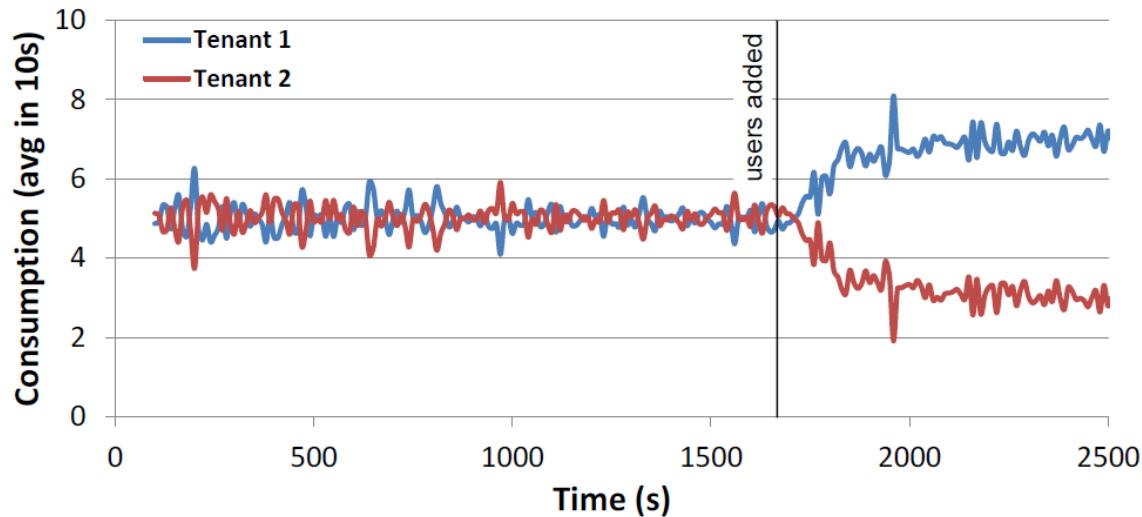
Indices:

$t$  = tenant

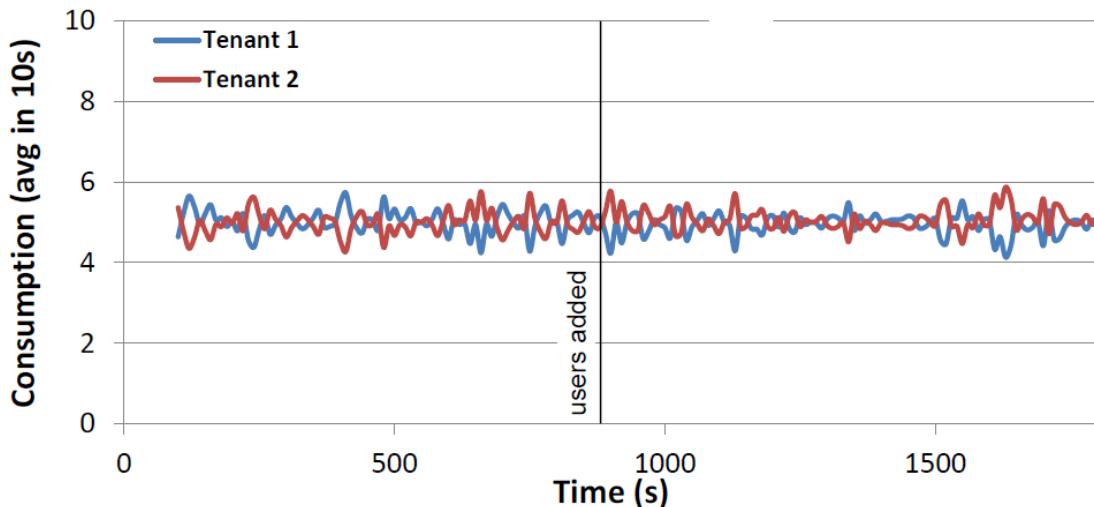
$r$  = request type

$i$  = resource

# Resource Isolation

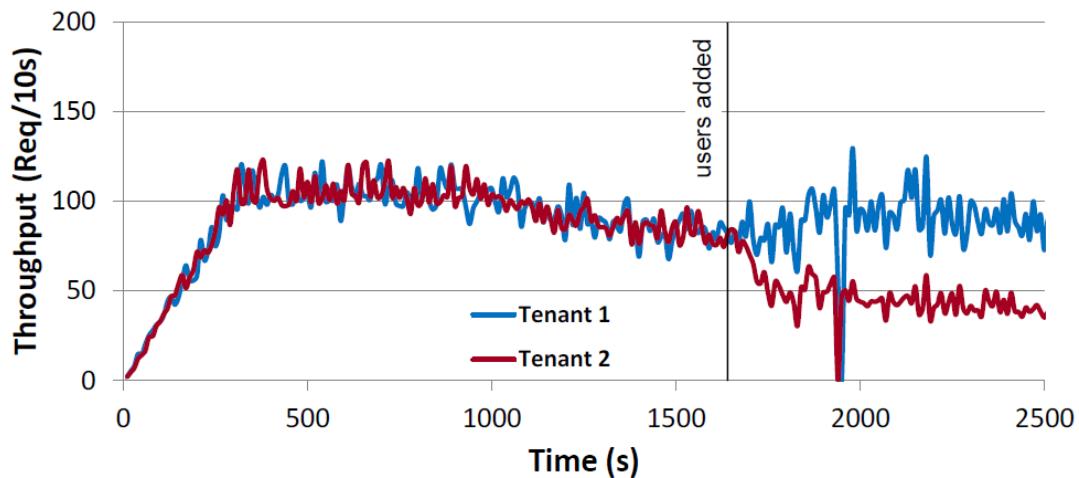


(c) Q2: Non-Isolated System Resource Consumption

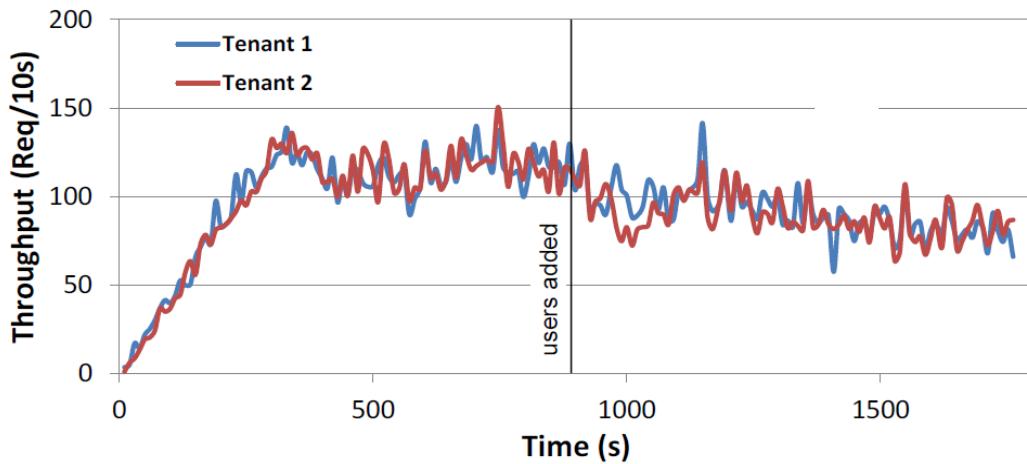


(f) Q2: Isolated System Resource Consumption

# Performance Isolation



(b) Q1: Non-Isolated System Throughput



(e) Q1: Isolated System Throughput

- Goal: Automatic vertical CPU scaling of VMs
- Zimbra is a collaboration server
- Transactional workload
  - SLA: Mails need to be delivered within 2 minutes
  - Mails may be queued

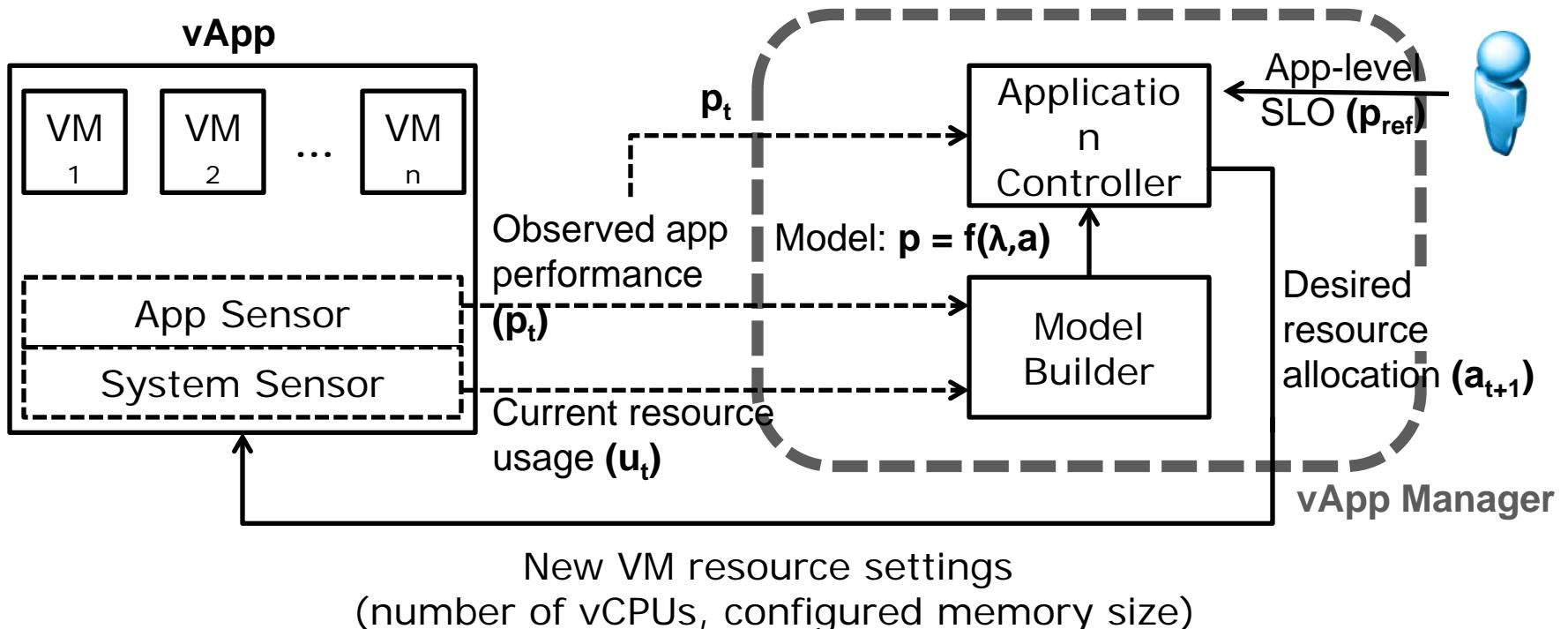
## IEEE SASO 2014.

### Runtime Vertical Scaling of Virtualized Applications via Online Model Estimation

Simon Spinner and Samuel Kounev  
University of Würzburg  
Email: {simon.spinner, samuel.kounev}@uni-wuerzburg.de

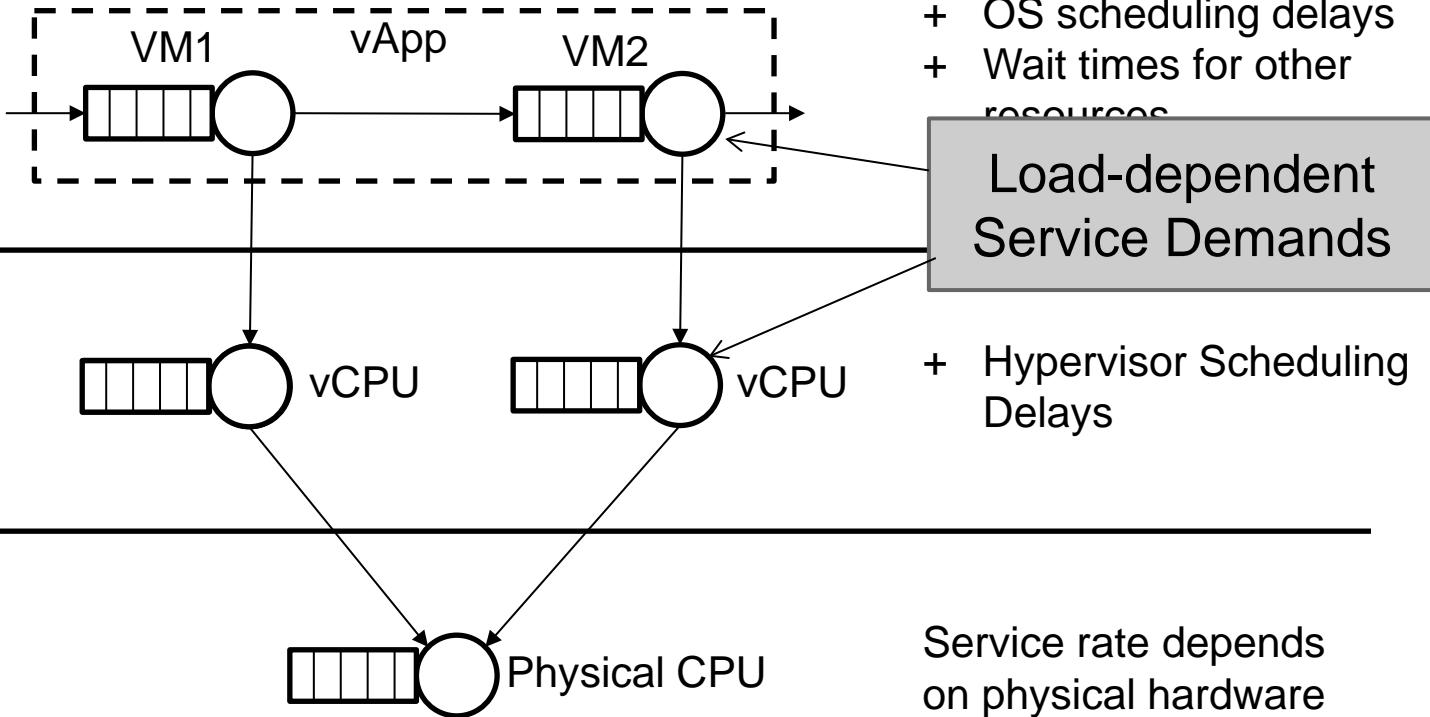
Xiaoyun Zhu, Lei Lu, Mustafa Uysal,  
Anne Holler and Rean Griffith, VMware, Inc.  
Email: {xzhu, llei, muysal, anne, rean}@vmware.com

# Approach Overview



# Layered Performance Model

*Application  
layer*

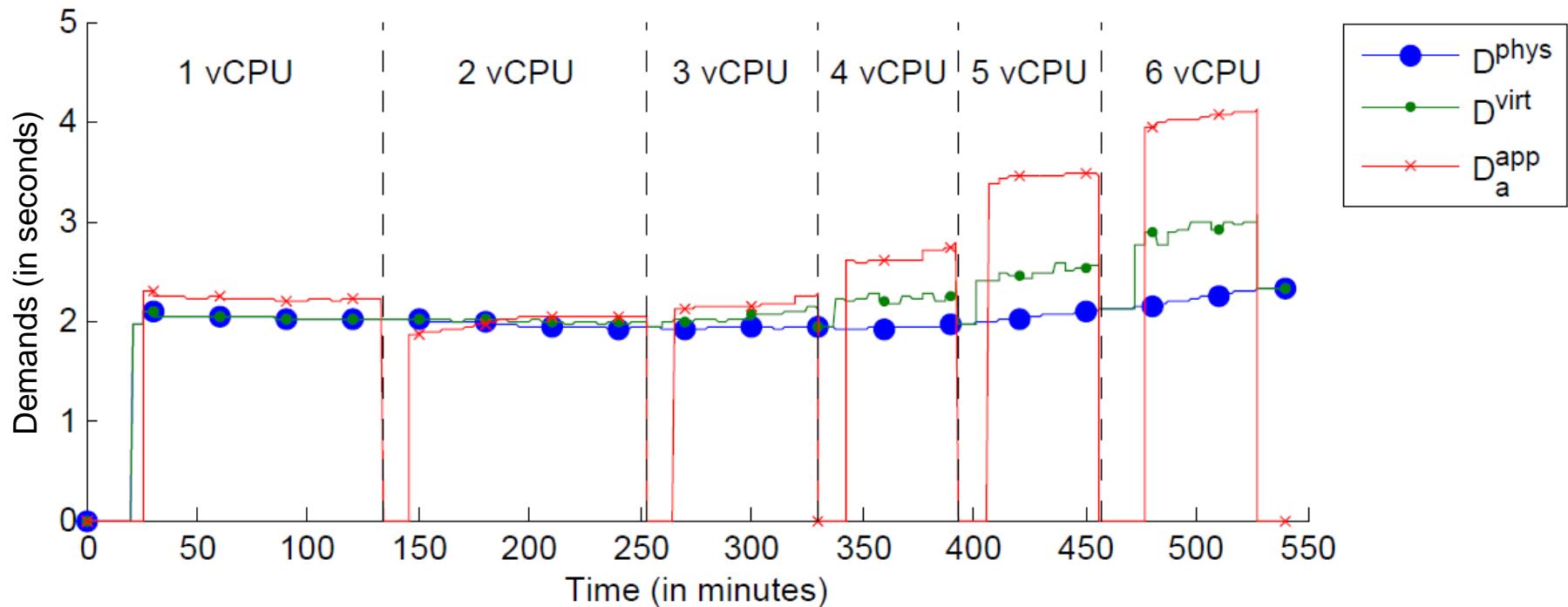


Hierarchical modeling approach (Method of Layers [1]):

Service time at layer  $i$  is equal to response time of an underlying closed queueing network at layer  $i - 1$

# Influence of Layers

Zimbra MTA with linearly increasing workload:



Estimated demands reflect contention at hypervisor and application level

# Reconfigurations

Controller <i>Zimbra MTA VM:</i>	Mean latency [s]	Reconfigurations	Mean vCPUs	Max vCPUs
Model-based	20.48	13	1.4	2
Trigger-based (1 min)	10.82	273	1.83	3
Trigger-based (5 min)	25.97	72	1.46	3
Static allocation	1385	0	1	1



**Model-based controller needs less reconfigurations and resources**

# Bibliography

Menascé, D. A. (2008). "Computing missing service demand parameters for performance models". In: CMG Conference Proceedings, pp. 241–248

Liu, Z., L. Wynter, C. H. Xia, and F. Zhang (2006). "Parameter inference of queueing models for IT systems using end-to-end measurements". In: Perform. Eval. 63.1, pp. 36–60

Kumar, D., A. N. Tantawi, and L. Zhang (2009a). "Real-time performance modeling for adaptive software systems with multi-class workload". In: Proceedings of the 17<sup>th</sup> Annual Meeting of the IEEE/ACM International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems, MASCOTS, pp. 1–4

Zheng, T., C. M. Woodside, and M. Litoiu (2008). "Performance Model Estimation and Tracking Using Optimal Filters". In: IEEE Trans. Software Eng. 34.3, pp. 391–406

# Bibliography

Wang, W., X. Huang, X. Qin, W. Zhang, J. Wei, and H. Zhong (2012). "Application-Level CPU Consumption Estimation: Towards Performance Isolation of Multi-tenancy Web Applications". In: Proceedings of the 2012 IEEE Fifth International Conference on Cloud Computing, CLOUD, pp. 439–446

Brosig, F., S. Kounev, and K. Krognann (2009). "Automated extraction of palladio component models from running enterprise Java applications". In: Proceedings of the 4th International Conference on Performance Evaluation Methodologies and Tools, VALUETOOLS, p. 10

Rolia, J. and V. Vetland (1995). "Parameter estimation for performance models of distributed application systems". In: Proceedings of the 1995 Conference of the Centre for Advanced Studies on Collaborative Research, CASCON, p. 54

# Bibliography

Kraft, S., S. Pacheco-Sanchez, G. Casale, and S. Dawson (2009). "Estimating service resource consumption from response time measurements". In: Proceedings of the 4th International Conference on Performance Evaluation Methodologies and Tools, VALUETOOLS, p. 48

Wang, W. and G. Casale (2013). "Bayesian Service Demand Estimation Using Gibbs Sampling". In: Proceedings of the 2013 IEEE 21st International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems, MASCOTS, pp. 567–576

G. Casale, P. Cremonesi, R. Turrin, Robust Workload Estimation in Queueing Network Performance Models, in: 16th Euromicro Conference on Parallel, Distributed and Network-Based Processing (PDP), 2008, pp. 183-187.

# Bibliography

P. Cremonesi, K. Dhyani, A. Sansottera, Service Time Estimation with a Refinement Enhanced Hybrid Clustering Algorithm, in: Analytical and Stochastic Modeling Techniques and Applications, Vol. 6148 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2010, pp. 291--305

P. Cremonesi, A. Sansottera, Indirect estimation of service demands in the presence of structural changes, Performance Evaluation 73 (0) (2014) 18--40, special Issue on the 9th International Conference on Quantitative Evaluation of Systems

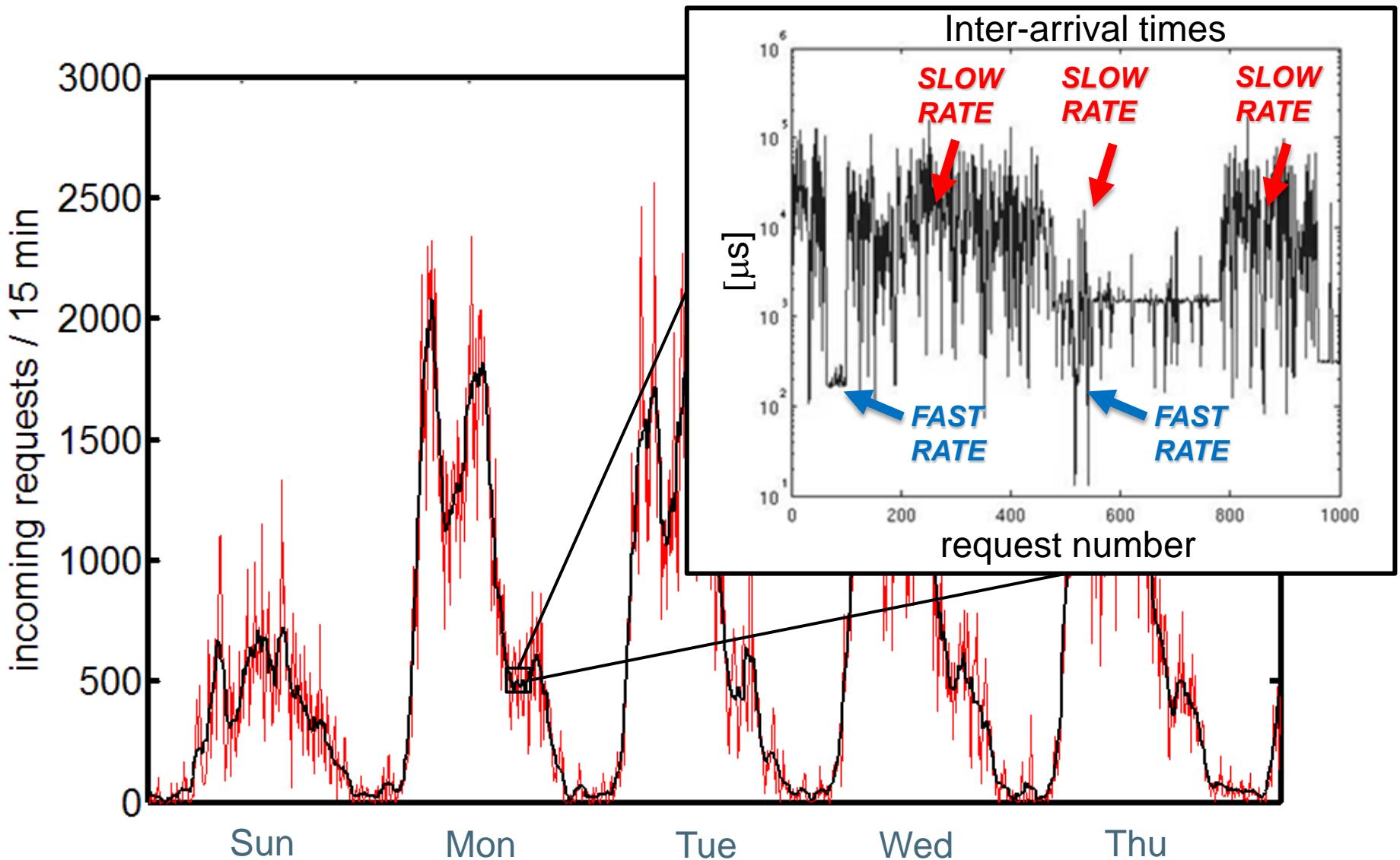
# Arrival Process Fitting

*Joint work with A. Sansottera and P. Cremonesi (DEIB, Politecnico di Milano, Italy)*

# Outline

- Introduction
- Moments and probabilities in Marked MAPs
- Fitting of second-order acyclic Marked MAPs
- Results
- Conclusions

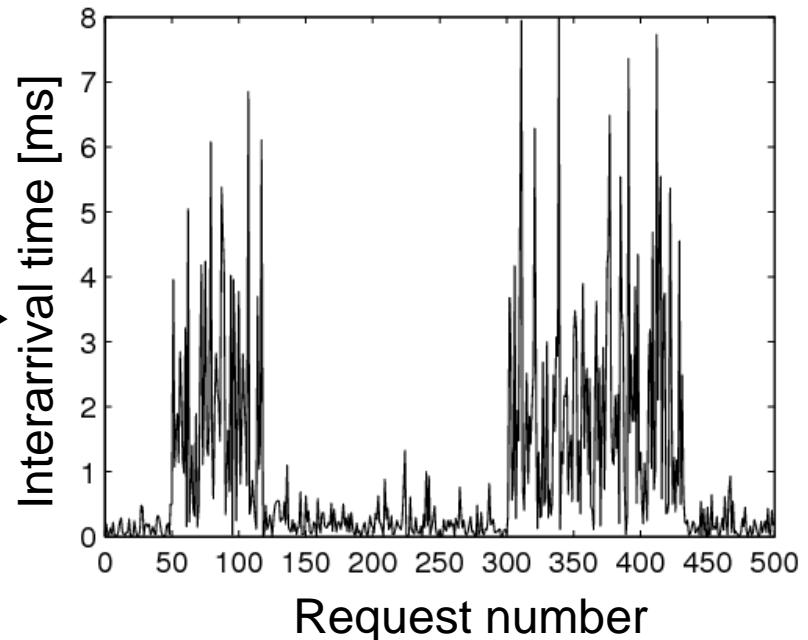
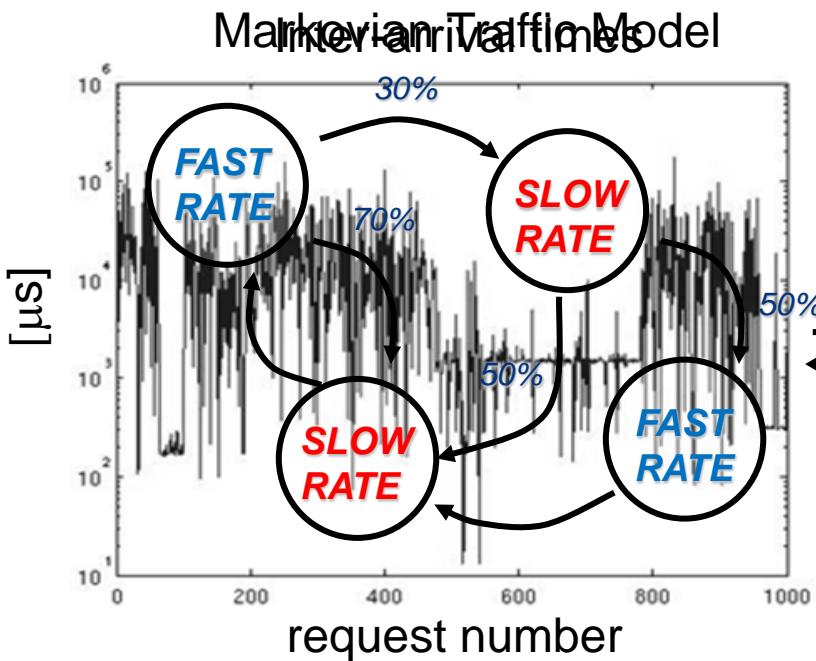
# Requests Traffic



# Arrival Process Modelling

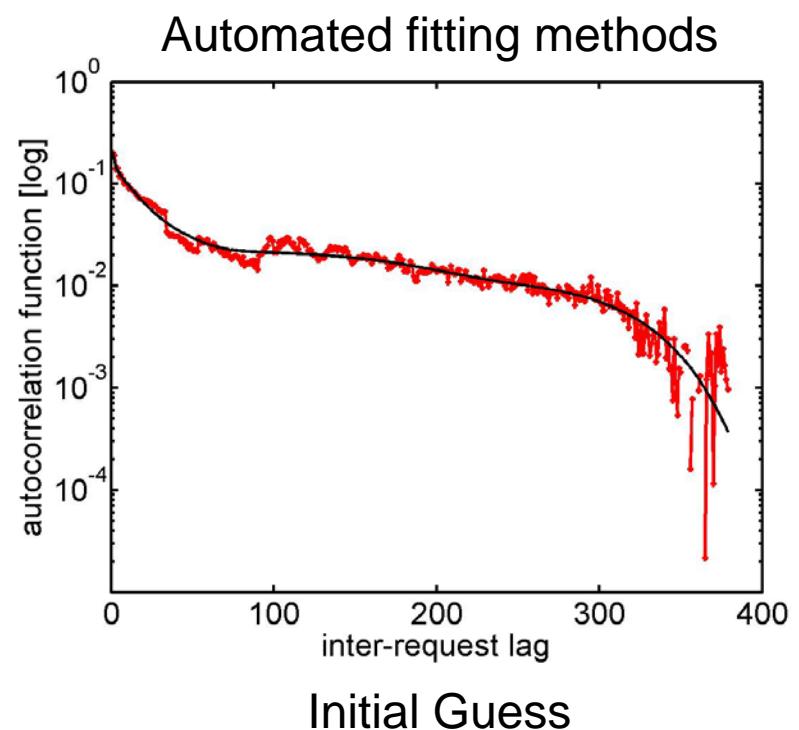
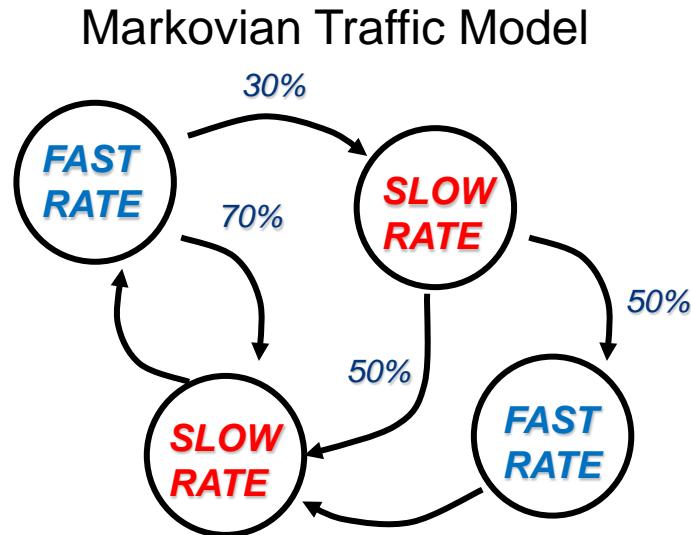
## ■ Stochastic models

- Generate statistically similar request arrival patterns
- Analytical models accelerate search for optimal decisions



## ■ Stochastic models

- Generate statistically similar request arrival patterns
- Analytical models accelerate search for optimal decisions



# Traffic Decomposition for QN

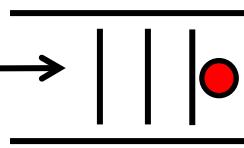
## ■ Network of queues

- mathematical abstraction for prediction, what-if scenarios, ...
- describes billions of possible states for the resources
- efficient output analysis techniques [Smirni, QEST'09]

Incoming  
Requests

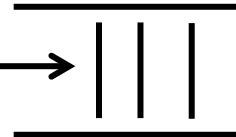


Web Server



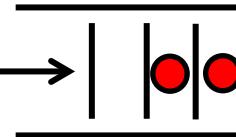
CPU

Database



CPUs

Storage



Disks

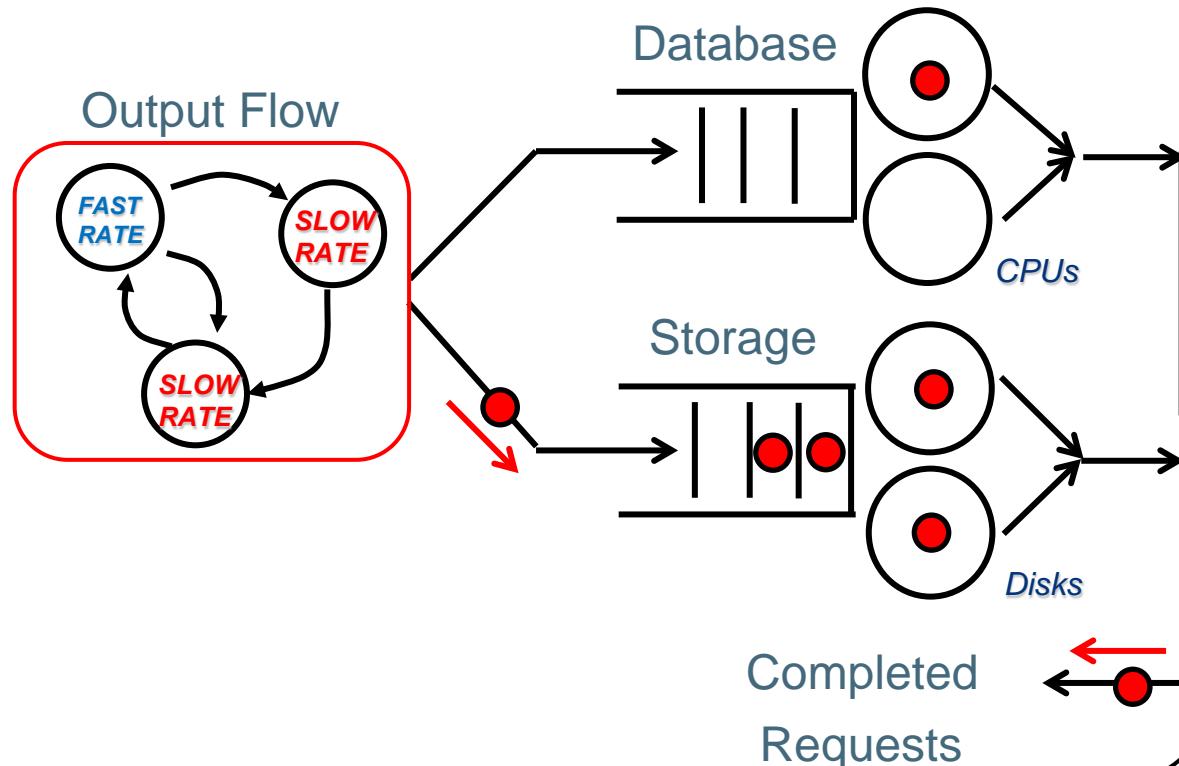
Cloud  
Resources

Completed  
Requests

# Traffic Decomposition for QN

## ■ Network of queues

- mathematical abstraction for prediction, what-if scenarios, ...
- describes billions of possible states for the resources
- developed efficient analysis techniques



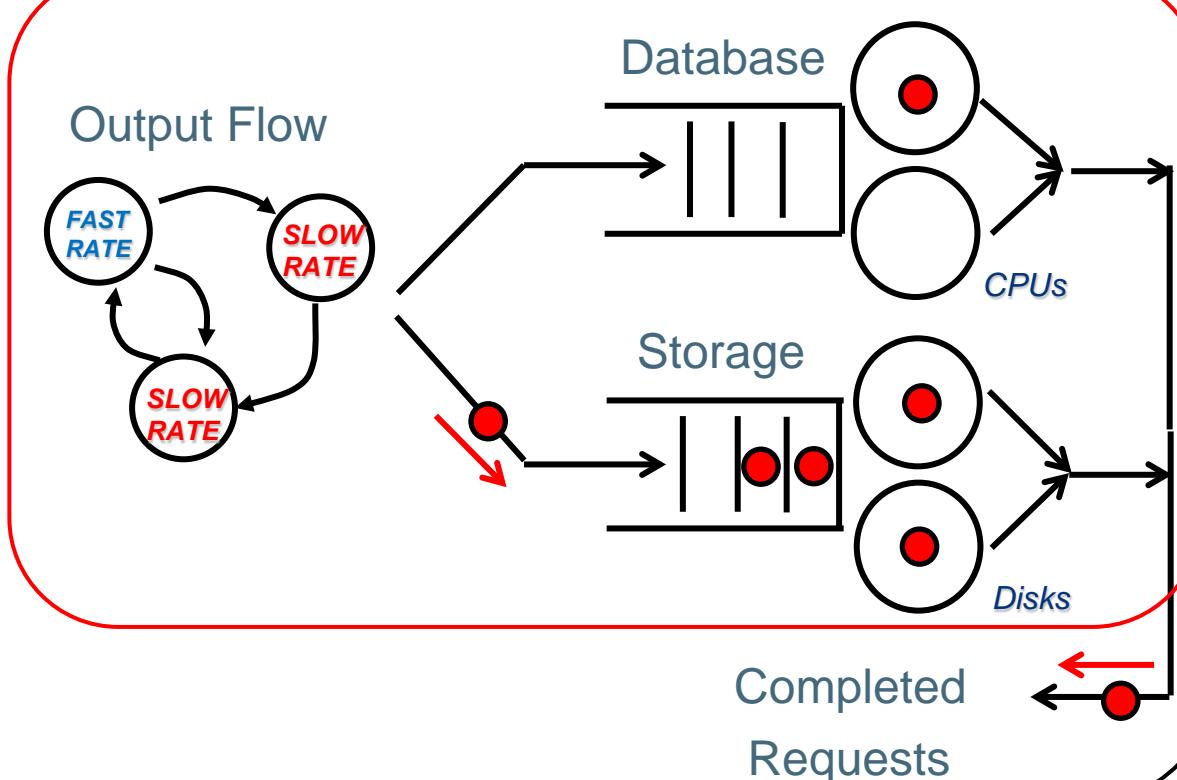
Cloud  
Resources

Completed  
Requests

# Traffic Decomposition for QN

## ■ Network of queues

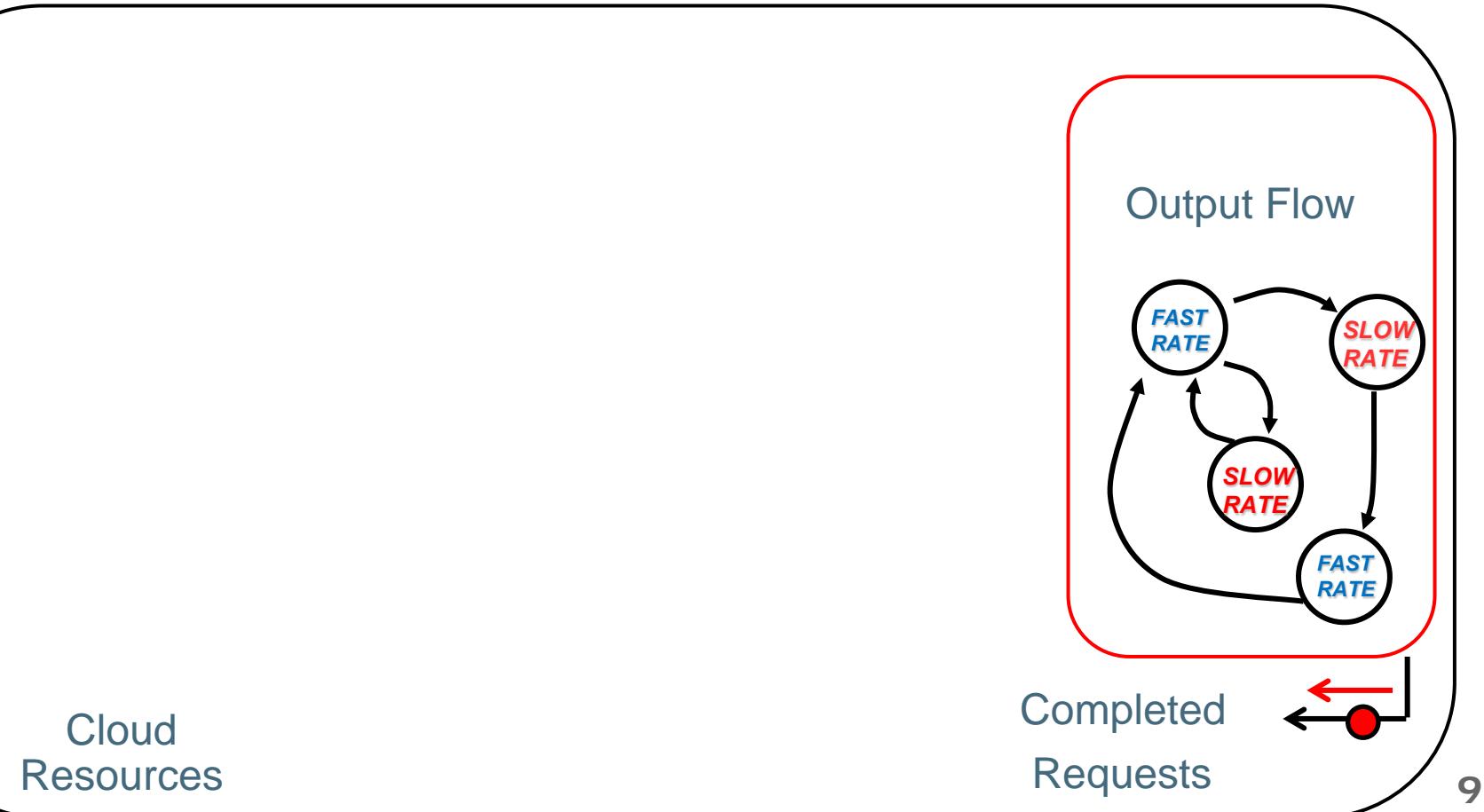
- mathematical abstraction for prediction, what-if scenarios, ...
- describes billions of possible states for the resources
- developed efficient analysis techniques



# Traffic Decomposition for QN

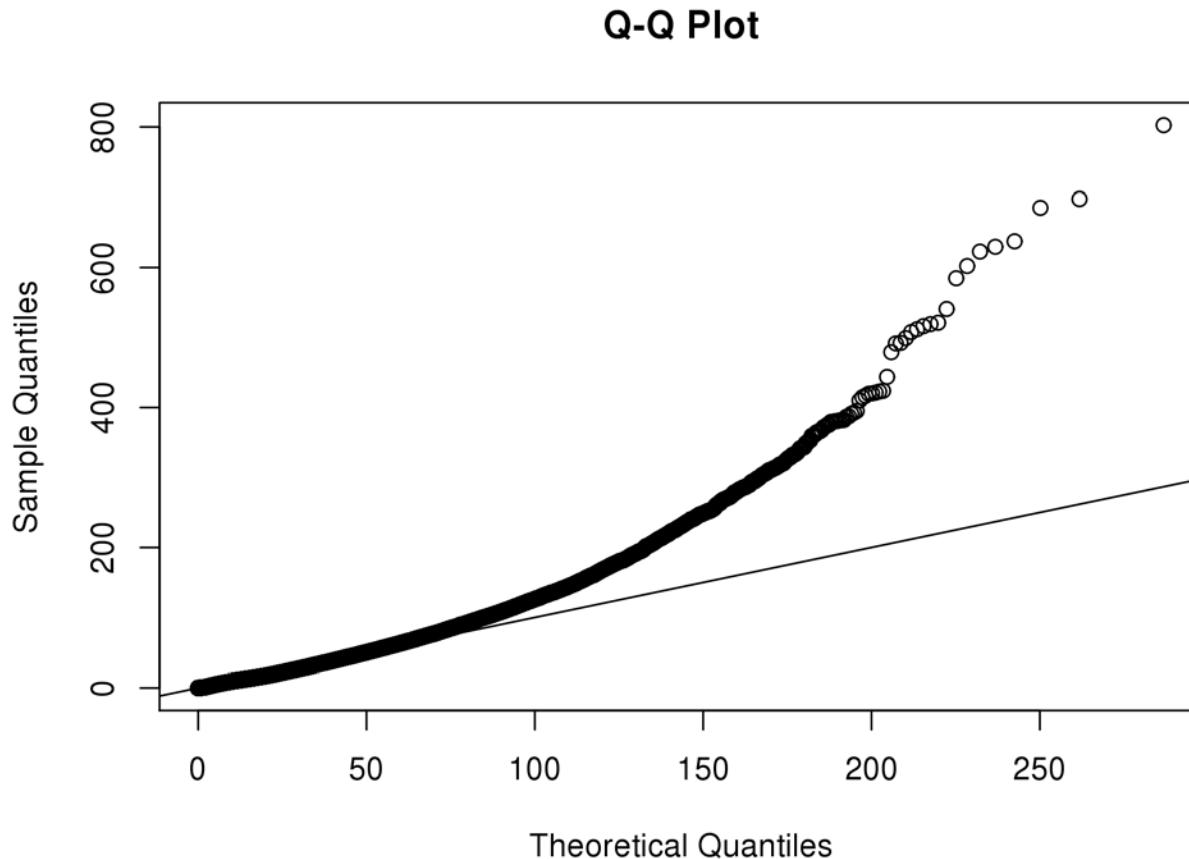
## ■ Network of queues

- mathematical abstraction for prediction, what-if scenarios, ...
- describes billions of possible states for the resources
- developed efficient analysis techniques

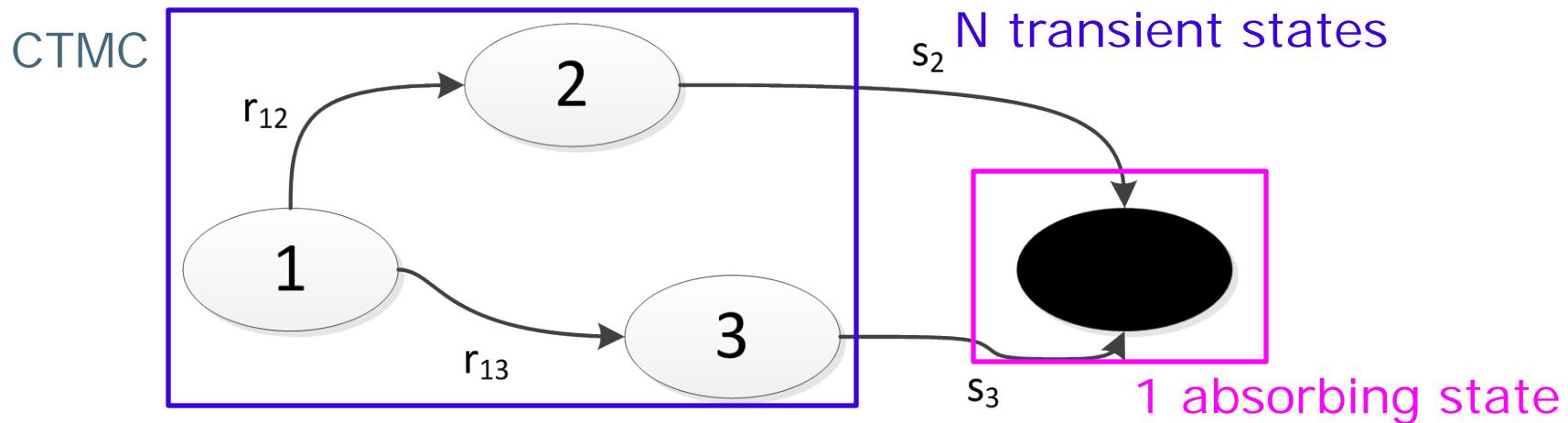


# Non-Poisson Arrivals

Microsoft Live Maps Back End Trace  
Disk read/write inter-issue time  
Poisson → Phase-type Renewal Processes



# PH-type Distribution



Phase-type Distribution: distribution of the time to absorption

$$Q = \begin{bmatrix} -r_{12} - r_{13} & r_{12} & r_{13} \\ 0 & -s_2 & 0 \\ 0 & 0 & -s_3 \end{bmatrix}$$

$$\begin{bmatrix} 0 \\ s_2 \\ s_3 \\ 0 \end{bmatrix}$$

$$\pi(0) = [\alpha_1 \quad \alpha_2 \quad \alpha_3 \quad 0]$$

Exit vector

$$D_0 = \begin{bmatrix} -r_{12} - r_{13} & r_{12} & r_{13} \\ 0 & -s_2 & 0 \\ 0 & 0 & -s_3 \end{bmatrix}$$

$$\alpha = [\alpha_1 \quad \alpha_2 \quad \alpha_3]$$

No-mass at 0 assumption Representation PH( $D_0, \alpha$ )

# PH-type Renewal Process

- Renewal Process with Phase-type distribution
- Inter-arrival times: i.i.d. with  $\text{PH}(D_0, \alpha)$  distribution
- Counting process  $N(t)$  is a CTMC
- Blocks of  $N$  states
- Block  $k$ :  $N(t) = k$
- After absorption, go to block  $k+1$
- Initial state in block  $k+1$ : probability  $\alpha$
- Rate of exit from state  $i$  and restart from state  $j$  =  $s_i \alpha_j$

Transitions in block  $k$  Event: exit from block  $k$ , restart from block  $k+1$

$$Q = \begin{bmatrix} D_0 & s \cdot \alpha & 0 & 0 \\ 0 & D_0 & s \cdot \alpha & 0 \\ 0 & 0 & D_0 & s \cdot \alpha \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \quad \begin{array}{l} N(t) = 1 \\ N(t) = 2 \\ N(t) = 3 \end{array}$$

# Some Tools for PH Fitting

## □ EMph (1996)

<http://home.imf.au.dk/asmus/pspapers.html>

- EM algorithm for ML fitting, based on Runge-Kutta methods
- Local optimization technique

## □ jPhase (2006)

<http://copa.uniandes.edu.co/software/jmarkov/index.html>

- Java library
- ML and canonical form fitting algorithms

## □ PhFit (2002)

<http://webspn.hit.bme.hu/~telek/tools.htm>

- Separate fit of distribution body and tail
- Both continuous and discrete ML distributions

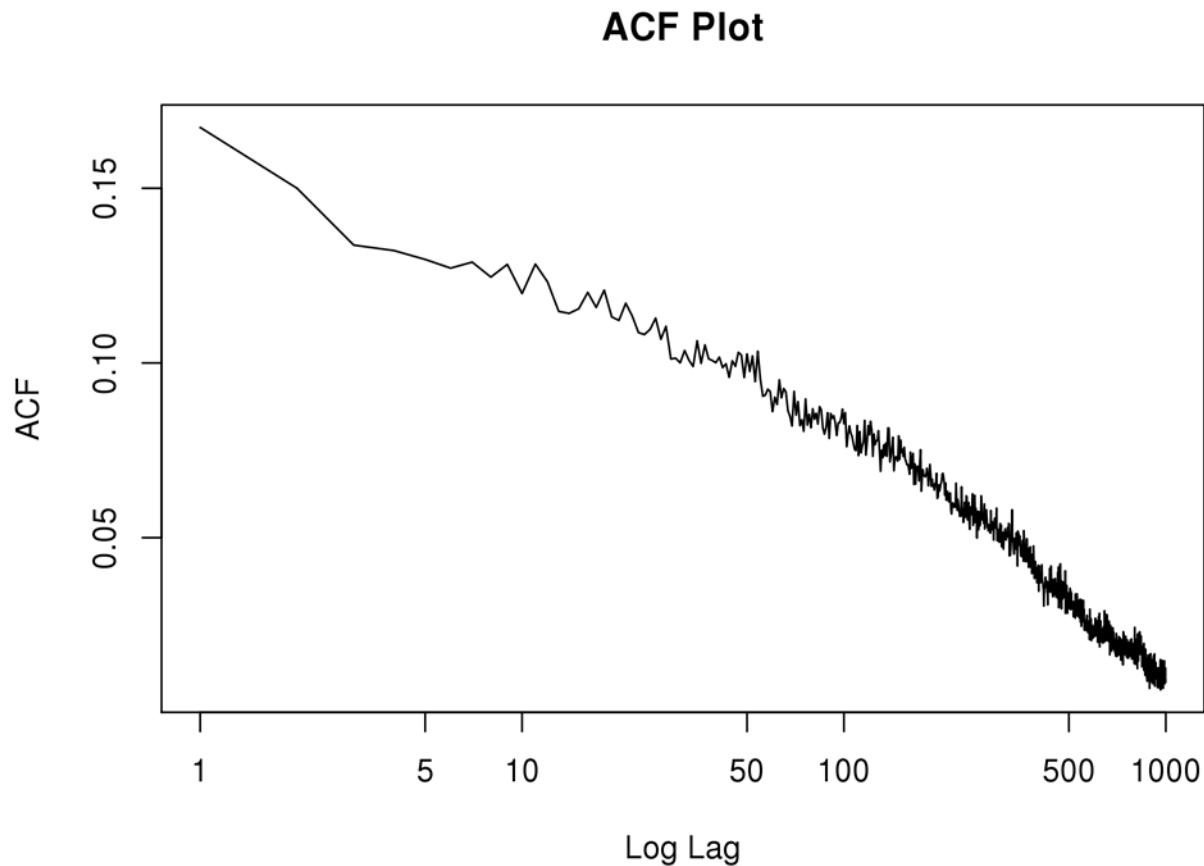
## □ G-FIT (2007)

<http://ls4-www.cs.uni-dortmund.de/home/thummler/gfit.tgz>

- Hyper-Erlang PHs used as building block
- Automatic aggregation of large traces, dramatic speed-up of computational times compared to EMph

# Correlated Arrivals

Microsoft Live Maps Back End Trace  
Disk read/write inter-issue time  
Phase-type Renewal Processes → Markovian Arrival Processes



# Markovian Arrival Process

## □ Phase-type Renewal Process

- Rate of exit from **state i** and restart from **state j** =  $s_i \mathbf{a}_j$

## □ Markovian Arrival Process (MAP)

- Rate of exit from **state i** and restart from **state j** =  $s_{ij}$

- Generalization of PH-Renewal: allows to model correlation

$$Q = \begin{bmatrix} D_0 & D_1 & 0 & 0 \\ 0 & D_0 & D_1 & 0 \\ 0 & 0 & D_0 & D_1 \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Representation: MAP( $D_0, D_1$ )  
Interval-stationary initialization

$$D_0 = \begin{bmatrix} -\lambda_1 & r_{12} & r_{13} & \cdots \\ r_{21} & -\lambda_2 & r_{23} & \cdots \\ \vdots & \vdots & \ddots & \vdots \\ r_{n1} & r_{n2} & \cdots & -\lambda_n \end{bmatrix}$$
$$D_1 = \begin{bmatrix} s_{11} & s_{12} & s_{13} & \cdots \\ s_{21} & s_{22} & s_{23} & \cdots \\ \vdots & \vdots & \ddots & \vdots \\ s_{n1} & s_{n2} & \cdots & -s_{nn} \end{bmatrix}$$

# Tools for MAP fitting

## ❑ KPC-Toolbox (2008)

<http://www.cs.wm.edu/MAPQN/kptoolbox.html>

- Moment-matching method
- Composition of large MAPs by two-state MAPs

$$\left. \begin{array}{l} MAP_a = \{\mathbf{D}_0^a, \mathbf{D}_1^a\} \\ MAP_b = \{\mathbf{D}_0^b, \mathbf{D}_1^b\} \end{array} \right\} \text{KPC Process} \quad MAP_a \otimes MAP_b = \{-\mathbf{D}_0^a \otimes \mathbf{D}_0^b, \mathbf{D}_1^a \otimes \mathbf{D}_1^b\}$$

- Property of KPC Process (similar relations for higher-order moments, ACF, ...)

$$E[X^k] = E_a[X^k]E_b[X^k]/k!$$

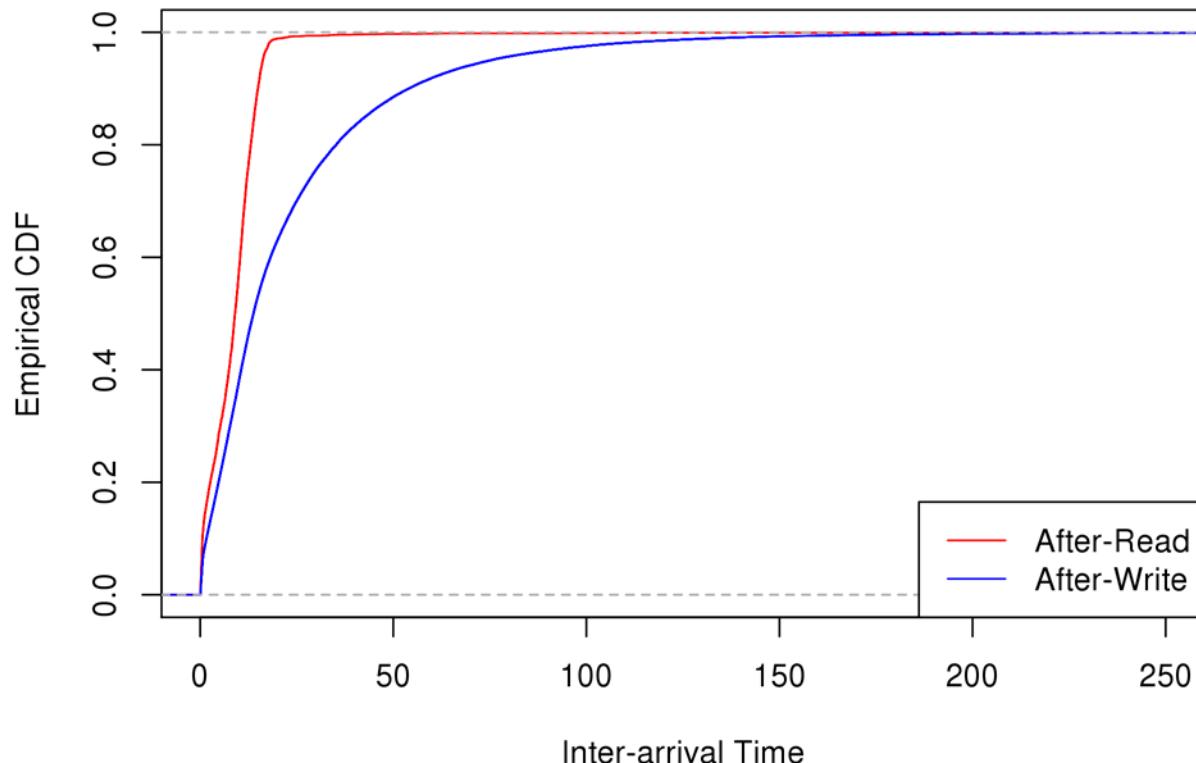
- Marked Markovian Arrival Processes (MMAPs)
  - Generalization of MAPs to model **multi-class arrivals**
  - Allow to model non-Poisson **cross-correlated** arrivals
  - Allow **efficient solution** of the models with matrix-analytic methods
  
- Modeling the arrival process at a queuing system (MMAP[K]/PH[k]/1-FCFS queue)
  - FCFS queues can be analyzed analytically using age process
  - Q-MAM: <https://bitbucket.org/qmam/qmam/src>
  - BU-Tools: <http://webspn.hit.bme.hu/~telek/tools/butools/>

# Multi-class Arrivals

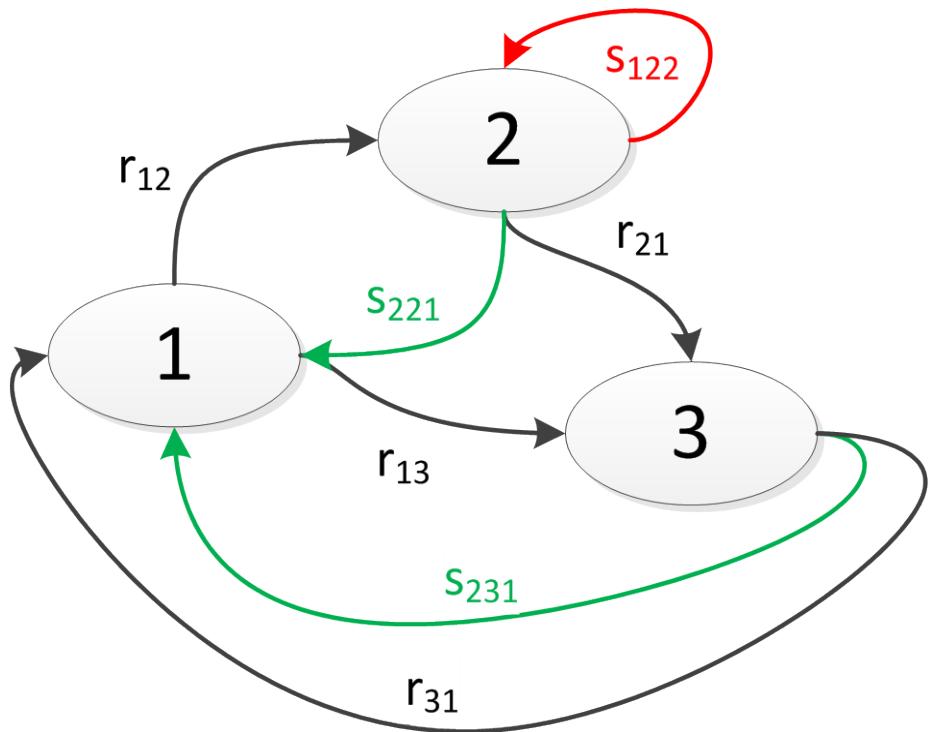
Microsoft Live Maps Back End Trace  
Disk read/write inter-issue time

Markovian Arrival Processes → Marked Markovian Arrival Processes

Conditional Empirical CDF



# Marked MAPs



$$D_0 = \begin{bmatrix} -r_{12} - r_{13} & r_{12} & r_{13} \\ 0 & -r_{21} - s_{122} - s_{221} & r_{21} \\ r_{31} & 0 & -r_{31} - s_{231} \end{bmatrix}$$

$$D_{11} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & s_{122} & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad D_{12} = \begin{bmatrix} 0 & 0 & 0 \\ s_{221} & 0 & 0 \\ s_{231} & 0 & 0 \end{bmatrix}$$

$(D_0, D_{11}, D_{12})$  is a representation of a MMAP[2] process (2 classes)

$$D_1 = D_{11} + D_{12}$$

$(D_0, D_1)$  is a representation of the MAP underlying the MMAP

# Fitting

## □ Fitting problem

- Marked trace from a real system:  $(X_i, C_i) \rightarrow \text{MMAP}$
- Queues with arrivals that follow MMAP can be solved analytically

## □ Two families of methods

- Maximum-likelihood
- Matching moments (or other characteristics)

## □ We focus on moment matching

- More computationally efficient
- In real systems, easier to save moments than the whole trace

# Issues of moment matching

- Representation of MMAPs is not minimal
- Number of parameters >> Degrees of freedom
- Hard to obtain analytical fitting formulas for the parameters
  - Easy: Parameters -> Moments
  - Hard: **Moments -> Parameters**
  - Requires solving a non-linear system of equations in the general case
  - Non-linear least squares for MMAP fitting [Buchholz, 2010]

# Issues of moment matching

- **Feasibility:** given a number of states  $n$  for the MMAF, which values of the moments can be fitted exactly?
  - Related issue: how to perform **approximate fitting?**
- Which characteristics best capture the queueing behavior?
  - Caveat 1: not all characteristics have known analytical formulas
  - Caveat 2: inverting the analytical formulas might be harder for some characteristics

# Outline

- Introduction
- Moments and probabilities in Marked MAPs
- Fitting of second-order acyclic Marked MAPs
- Results

# Definitions

- Ordinary moment of order j:

$$M_j = E[X_i^j]$$



- Backward moment of order j for class c (green):

$$B_{j,c} = E[X_i^j | C_i = c]$$



- Forward moment of order j for class c (green):

$$F_{j,c} = E[X_i^j | C_{i-1} = c]$$



- Cross moments of order j for class c followed by class k:

$$M_{j,c,k} = E[X_i^j | C_{i-1} = c, C_i = k]$$



- Probability of a class c arrival:

$$p_c = \Pr[C_i = c]$$

- “Transition” probability of a class c arrival followed by class k

$$p_{c,k} = \Pr[C_{i-1} = c, C_i = k]$$

# Moment Dependencies

- Ordinary moments can be expressed as **linear combination** of
  - the forward moments, weighted by the class probabilities
  - the backward moments, weighted by the class probabilities
  - the cross moments, weighted by the class-transition probabilities

$$M_j = \sum_{c=1}^m p_c B_{j,c} = \sum_{c=1}^m p_c F_{j,c} = \sum_{c=1}^m \sum_{k=1}^m p_{c,k} M_{j,c,k}$$

$$\sum_{c=1}^m p_c = 1$$

$$\sum_{k=1}^m p_{c,k} = p_c$$

$$\sum_{c=1}^m p_{c,k} = p_k$$

$$B_{j,c} = p_c^{-1} \sum_k p_{k,c} M_{j,k,c}$$

$$F_{j,c} = p_c^{-1} \sum_k p_{c,k} M_{j,c,k}$$

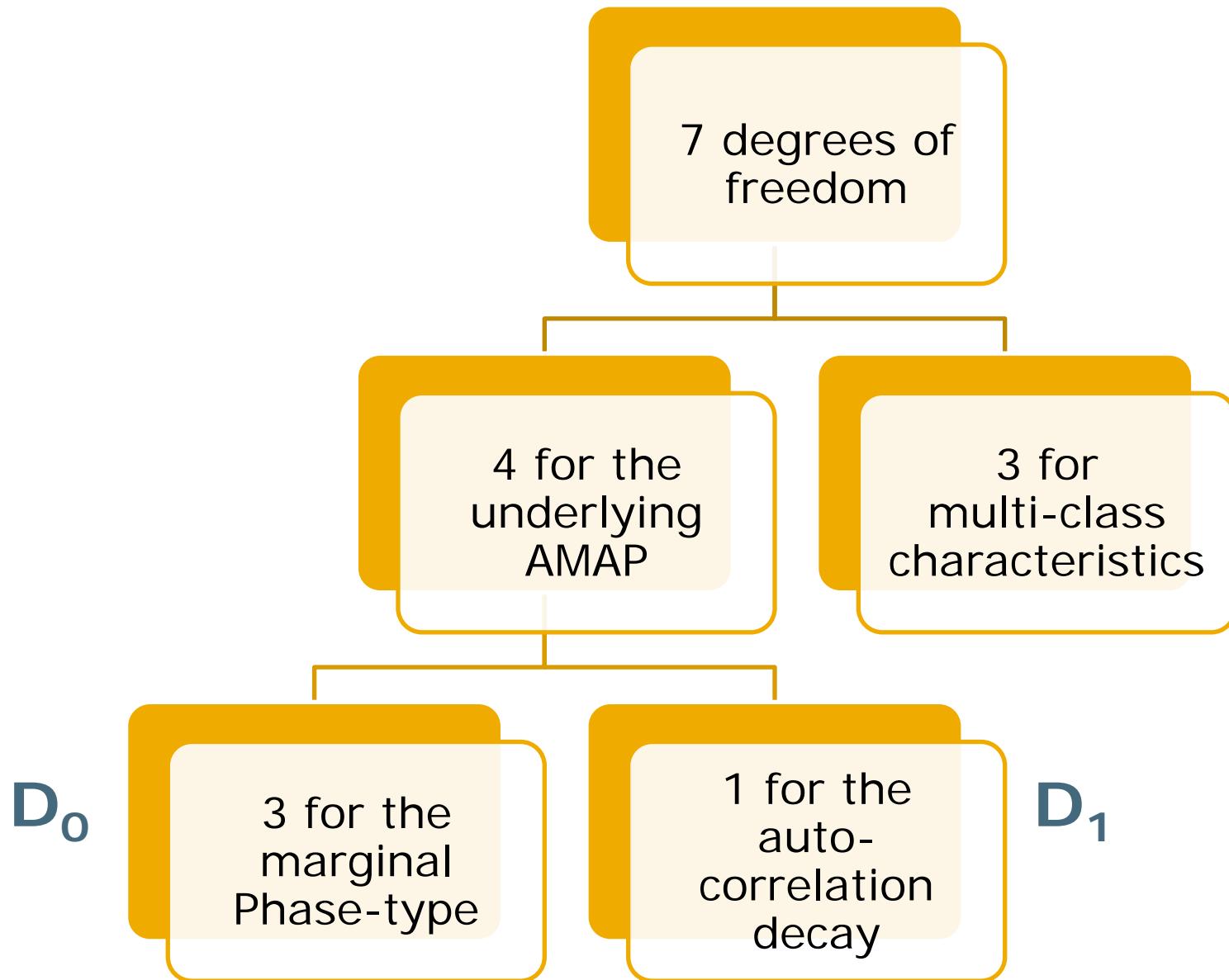
$$M_j = \sum_{c=1}^m \sum_{k=1}^m p_{c,k} M_{j,c,k}$$

For 2 classes and  $j = 1$   
 Linear system for  $M_{1ck}$   
 4 unknowns, rank 3

A cross-moment might  
 be needed to uniquely  
 determine a second-  
 order MMAP[2]

# Outline

- Introduction
- Moments and probabilities in Marked MAPs
- Fitting of second-order acyclic Marked MAPs
- Results

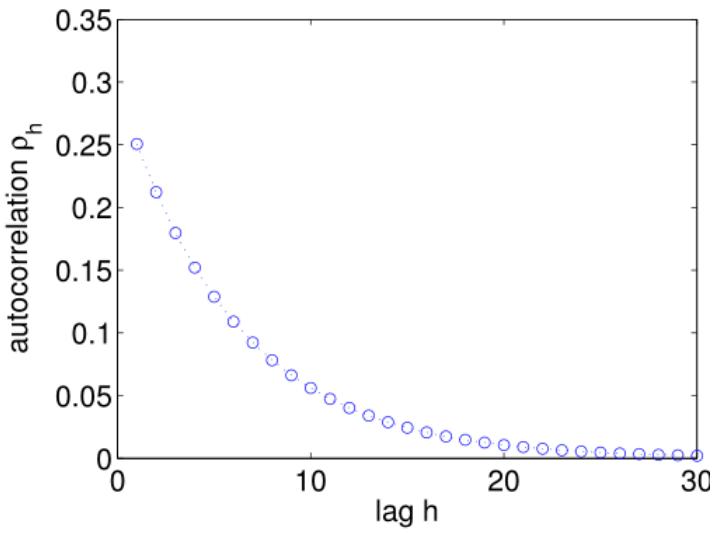


# AMMAP[2] Fitting

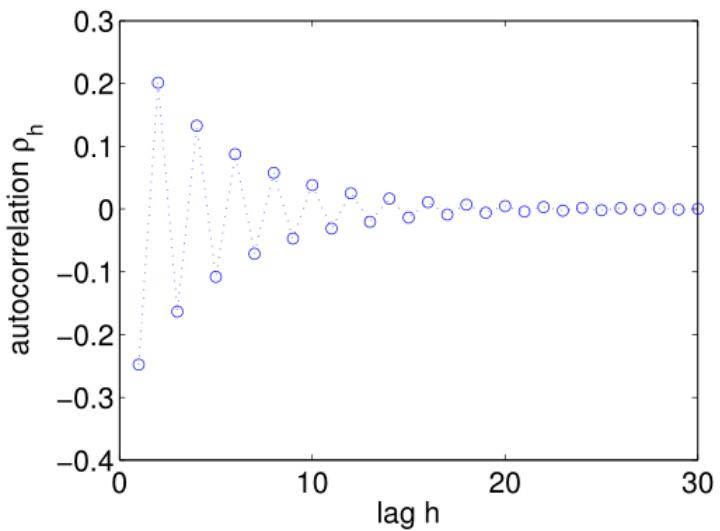
- Any MAP(2) has geometric autocorrelation decay with rate  $\gamma$ 
  - Canonical form for the underlying MAP(2) [Bodrog et al., 2010]
- Acyclic: two forms for  $\gamma > 0$  and  $\gamma < 0$
- For  $\gamma = 0$ , acyclic phase-type renewal

$$D_0 = \begin{bmatrix} -\frac{1}{h_1} & \frac{r_1}{h_1} \\ 0 & -\frac{1}{h_2} \end{bmatrix}$$

$\gamma > 0$



$\gamma < 0$



# AMMAP[2] Fitting

- ❑ Any MAP(2) has geometric autocorrelation decay with rate  $\gamma$ 
  - Canonical form for the underlying MAP(2) [Bodrog et al., 2010]
- Acyclic: two forms for  $\gamma > 0$  and  $\gamma < 0$
- For  $\gamma = 0$ , acyclic phase-type renewal

$$D_0 = \begin{bmatrix} -\frac{1}{h_1} & \frac{r_1}{h_1} \\ 0 & -\frac{1}{h_2} \end{bmatrix}$$

$\gamma > 0$

$\gamma < 0$

$$D_1 = \begin{bmatrix} \frac{(1-r_1)}{h_1} & \boxed{0} \\ \frac{(1-r_2)}{h_2} & \frac{r_2}{h_2} \end{bmatrix}$$

$$D_1 = \begin{bmatrix} \boxed{0} & \frac{(1-r_1)}{h_1} \\ \frac{(1-r_2)}{h_2} & \frac{r_2}{h_2} \end{bmatrix}$$

# AMMAP[2] Fitting

- Any MAP(2) has geometric autocorrelation decay with rate  $\gamma$
- Canonical form for the underlying MAP(2) [Bodrog et al., 2010]
- **Acyclic:** two forms for  $\gamma > 0$  and  $\gamma < 0$
- For  $\gamma = 0$ , acyclic phase-type renewal

$$D_0 = \begin{bmatrix} -\frac{1}{h_1} & \frac{r_1}{h_1} \\ 0 & -\frac{1}{h_2} \end{bmatrix}$$

$\gamma > 0$

$$D_{1,1} = \begin{bmatrix} \frac{q_{1,1}(1-r_1)}{h_1} & 0 \\ \frac{q_{2,1}(1-r_2)}{h_2} & \frac{q_{2,2}r_2}{h_2} \end{bmatrix}$$

$$D_{1,2} = \begin{bmatrix} \frac{(1-q_{1,1})(1-r_1)}{h_1} & 0 \\ \frac{(1-q_{2,1})(1-r_2)}{h_2} & \frac{(1-q_{2,2})r_2}{h_2} \end{bmatrix}$$

$\gamma < 0$

$$D_{1,1} = \begin{bmatrix} 0 & \frac{q_{1,1}(1-r_1)}{h_1} \\ \frac{q_{2,1}(1-r_2)}{h_2} & \frac{q_{2,2}r_2}{h_2} \end{bmatrix}$$

$$D_{1,2} = \begin{bmatrix} 0 & \frac{(1-q_{1,1})(1-r_1)}{h_1} \\ \frac{(1-q_{2,1})(1-r_2)}{h_2} & \frac{(1-q_{2,2})r_2}{h_2} \end{bmatrix}$$

3 degrees of freedom

3 degrees of freedom

- How to spend the 3 available degrees of freedom?
- We have found closed, analytical formulas for the three parameters  $q_{11}$ ,  $q_{21}$ ,  $q_{22}$ , for both canonical forms
- Three different sets of characteristics considered
  - **Class probabilities** and...
  - 1) **Forward** moments and **backward** moments
  - 2) **Forward** moments and **class transition** probabilities
  - 3) **Backward** moments and **class transition** probabilities

# AMMAP[m] Fitting

- How to handle more than 2 classes?

$$D_0 = \begin{bmatrix} -4 & 2 \\ 0 & -5 \end{bmatrix}$$

$$D_{1,1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$D_{1,2} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

$$D_{1,3} = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$$

$$p_1 = 0.29$$

$$F_{11} = 0.08$$

$$B_{11} = 0.08$$

$$p_2 = 0.43$$

$$F_{12} = 0.13$$

$$B_{12} = 0.12$$

$$p_3 = 0.29$$

$$F_{13} = 0.08$$

$$B_{13} = 0.09$$

$$D_0 = \begin{bmatrix} -4 & 2 \\ 0 & -5 \end{bmatrix}$$

$$D_{1,1}^1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$D_{1,2}^1 = \begin{bmatrix} 1 & 0 \\ 2 & 2 \end{bmatrix}$$

$$D_0 = \begin{bmatrix} -4 & 2 \\ 0 & -5 \end{bmatrix}$$

$$D_{1,1}^2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

$$D_{1,2}^2 = \begin{bmatrix} 1 & 0 \\ 1 & 2 \end{bmatrix}$$

$$D_0 = \begin{bmatrix} -4 & 2 \\ 0 & -5 \end{bmatrix}$$

$$D_{1,1}^3 = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$$

$$D_{1,2}^3 = \begin{bmatrix} 2 & 0 \\ 1 & 2 \end{bmatrix}$$

- Latest version:

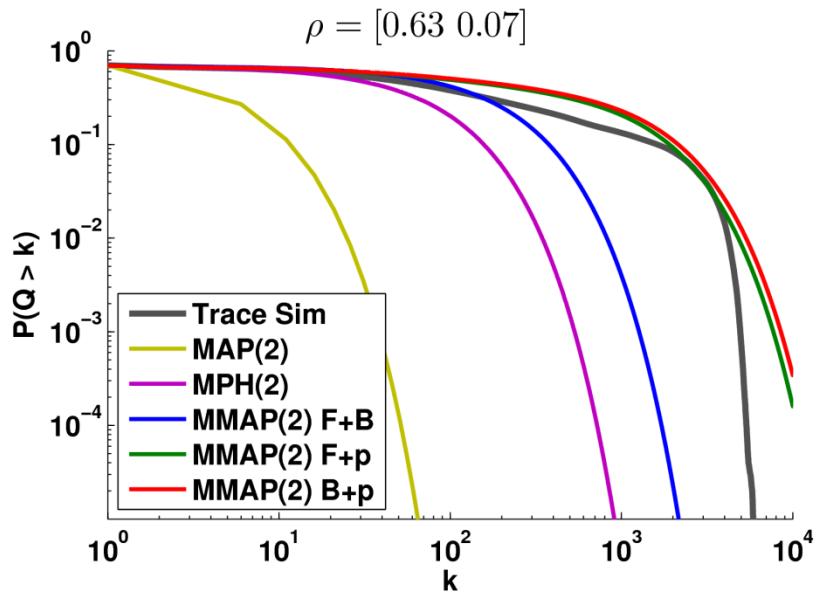
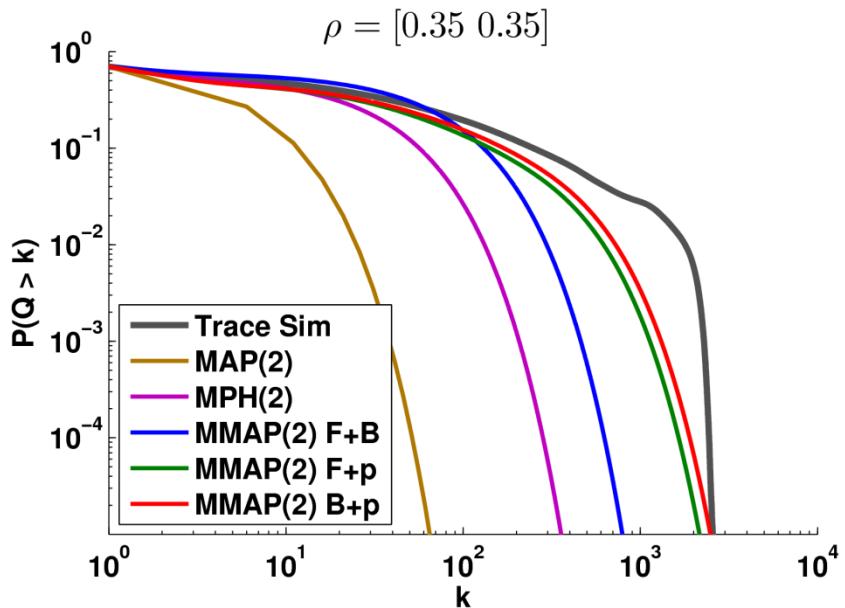
<https://github.com/Imperial-AESOP/M3A>

- A set of Matlab functions designed for computing the statistical descriptors of MMAPs and fitting marked traces with MMAPs
- Syntax compatibility with KPC-Toolbox
  - M3A's MMAPs are treated by KPC-Toolbox as MAPs

# Outline

- Introduction
- Moments and probabilities in Marked MAPs
- Fitting of second-order acyclic Marked MAPs
- **Results**

## Microsoft Live Maps Back End Trace – Disk read/write inter-issue time Simulation of $*/M/1$ Queue



# Bibliography

Menascé, D. A. (2008). "Computing missing service demand parameters for performance models". In: CMG Conference Proceedings, pp. 241–248

Liu, Z., L. Wynter, C. H. Xia, and F. Zhang (2006). "Parameter inference of queueing models for IT systems using end-to-end measurements". In: Perform. Eval. 63.1, pp. 36–60

Kumar, D., A. N. Tantawi, and L. Zhang (2009a). "Real-time performance modeling for adaptive software systems with multi-class workload". In: Proceedings of the 17<sup>th</sup> Annual Meeting of the IEEE/ACM International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems, MASCOTS, pp. 1–4

Zheng, T., C. M. Woodside, and M. Litoiu (2008). "Performance Model Estimation and Tracking Using Optimal Filters". In: IEEE Trans. Software Eng. 34.3, pp. 391–406

# Bibliography

Wang, W., X. Huang, X. Qin, W. Zhang, J. Wei, and H. Zhong (2012). "Application-Level CPU Consumption Estimation: Towards Performance Isolation of Multi-tenancy Web Applications". In: Proceedings of the 2012 IEEE Fifth International Conference on Cloud Computing, CLOUD, pp. 439–446

Brosig, F., S. Kounev, and K. Krognann (2009). "Automated extraction of palladio component models from running enterprise Java applications". In: Proceedings of the 4th International Conference on Performance Evaluation Methodologies and Tools, VALUETOOLS, p. 10

Rolia, J. and V. Vetland (1995). "Parameter estimation for performance models of distributed application systems". In: Proceedings of the 1995 Conference of the Centre for Advanced Studies on Collaborative Research, CASCON, p. 54

# Bibliography

Kraft, S., S. Pacheco-Sanchez, G. Casale, and S. Dawson (2009). "Estimating service resource consumption from response time measurements". In: Proceedings of the 4th International Conference on Performance Evaluation Methodologies and Tools, VALUETOOLS, p. 48

Wang, W. and G. Casale (2013). "Bayesian Service Demand Estimation Using Gibbs Sampling". In: Proceedings of the 2013 IEEE 21st International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems, MASCOTS, pp. 567–576

G. Casale, P. Cremonesi, R. Turrin, Robust Workload Estimation in Queueing Network Performance Models, in: 16th Euromicro Conference on Parallel, Distributed and Network-Based Processing (PDP), 2008, pp. 183-187.

# Bibliography

P. Cremonesi, K. Dhyani, A. Sansottera, Service Time Estimation with a Refinement Enhanced Hybrid Clustering Algorithm, in: Analytical and Stochastic Modeling Techniques and Applications, Vol. 6148 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2010, pp. 291--305

P. Cremonesi, A. Sansottera, Indirect estimation of service demands in the presence of structural changes, Performance Evaluation 73 (0) (2014) 18--40, special Issue on the 9th International Conference on Quantitative Evaluation of Systems

Giuliano Casale, Evgenia Smirni: KPC-toolbox: fitting Markovian arrival processes and phase-type distributions with MATLAB. SIGMETRICS Performance Evaluation Review 39(4): 47 (2012)

Andrea Sansottera, Giuliano Casale, Paolo Cremonesi: Fitting second-order acyclic Marked Markovian Arrival Processes. DSN 2013: 1-12