Detecting **Transient Bottlenecks** in n-Tier Applications through Fine-Grained Analysis

Qingyang Wang, Yasuhiko Kanemasa, Jack Li, Deepal Jayasinghe, Toshihiro Shimizu, Masazumi Matsubara, Motoyuki Kawaba, Calton Pu

Georgia Tech College of Computing cercs FUJITSU

Dilemma between Good Performance and High Utilization

- Low response time is essential for Quality of Service (e.g., SLA for web-facing e-commerce applications).
  - Amazon reports that every 100ms increase in the page load decreases sales by 1%.

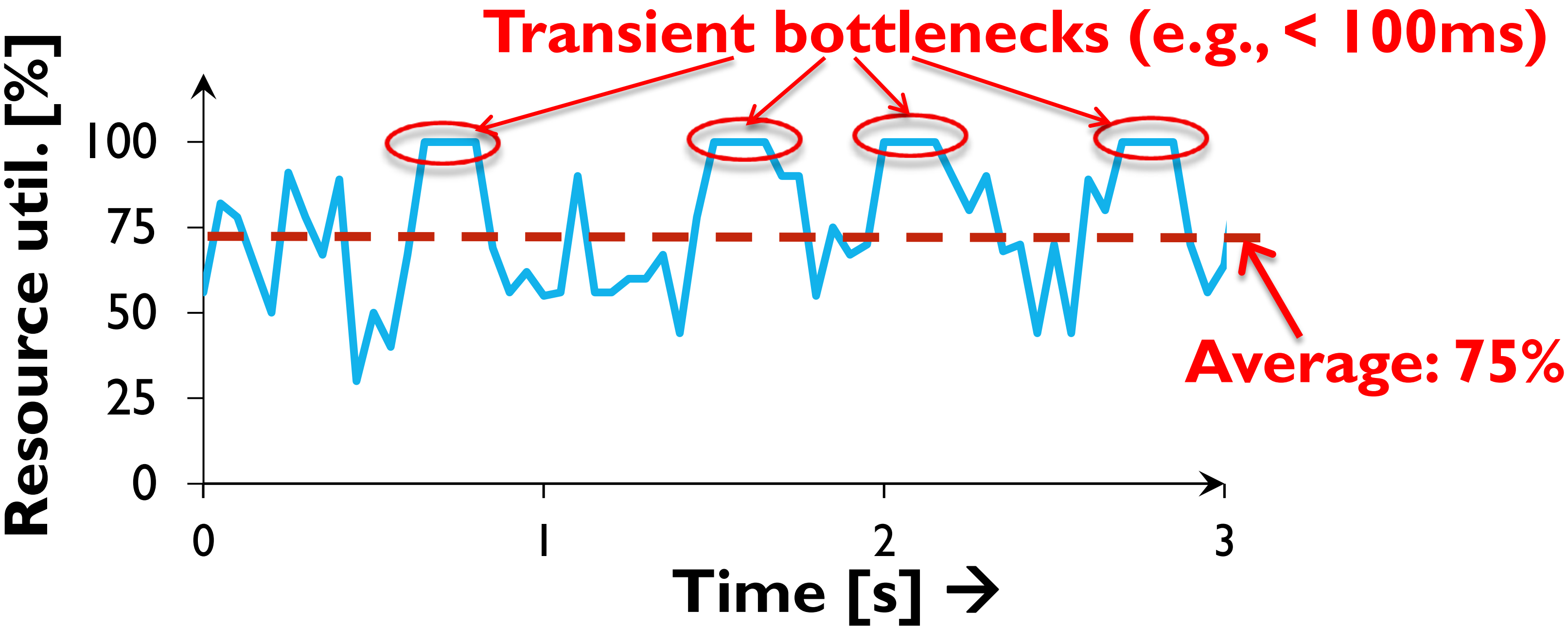
amazon web services™

- Achieving low response time at high resource utilization is challenging.
  - Servers in typical data centers are only busy 18% time on average, wasting power.

Gartner *[Dec 2010]*

An Important Factor: Transient Bottlenecks

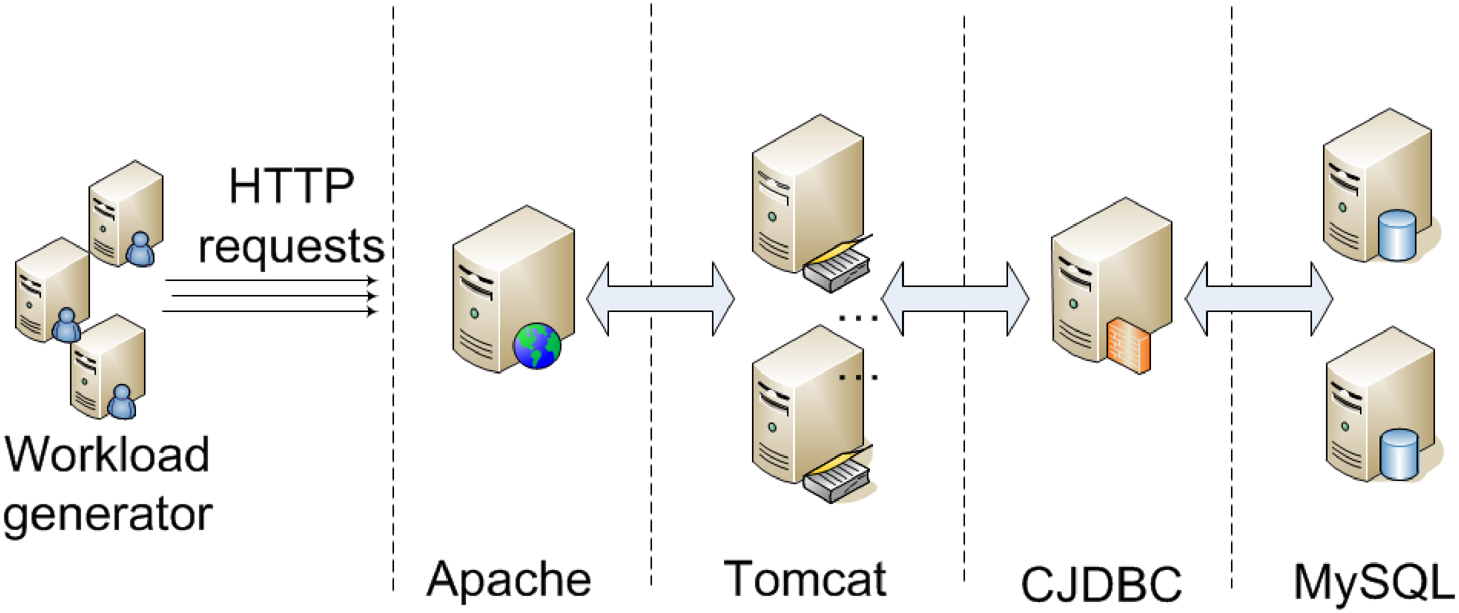
- Transient bottlenecks (e.g., ten of milliseconds) can cause wide-range response time variations for web-facing n-tier applications. *[ICAC'12]*


Transient bottlenecks (e.g., < 100ms)

Average: 75%

Challenging for typical monitoring tools with coarse granularity

Experimental Setup



- ❑ RUBBoS benchmark: a bulletin board system like Slashdot
- ❑ 24 web interactions

  CPU intensive
- ❑ Workload consists of emulated clients

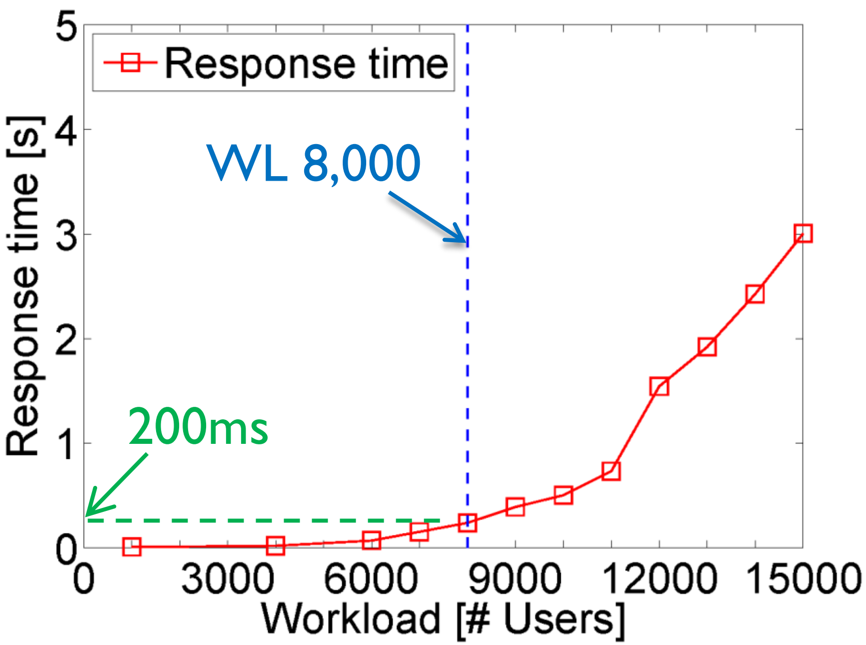
- ❑ Intel Xeon E5607

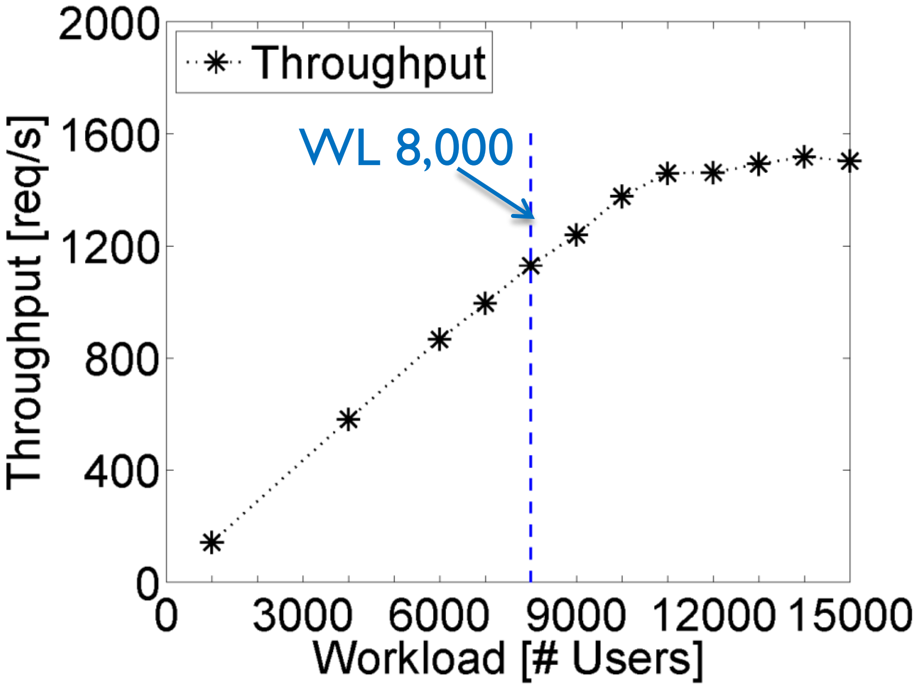
  2 quad-core 2.26 GHz

  16 GB memory

Motivational Example

- Response time & throughput of a 3-minute benchmark on the 4-tier application with increasing workloads.



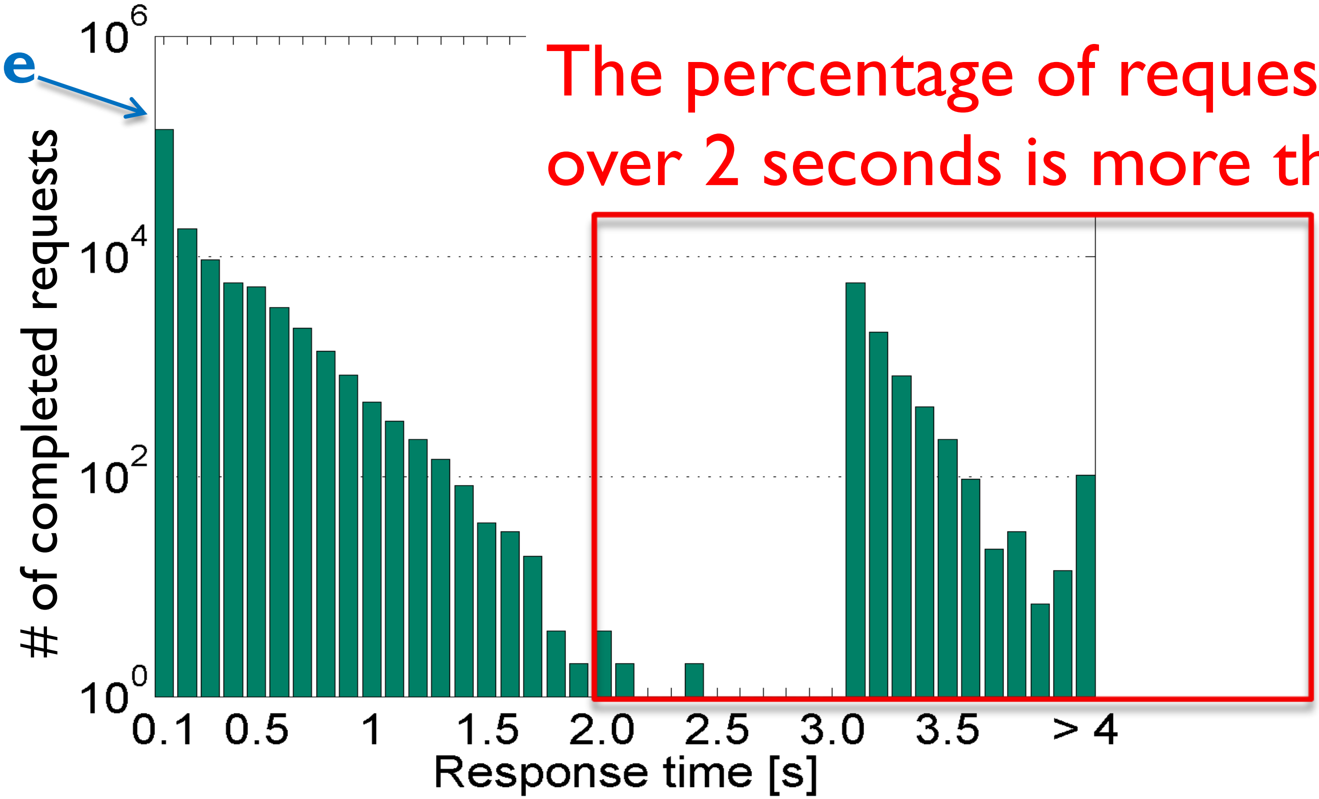


Average response time is low at workload 8,000, how about response time distribution?

Measured Long-Tail Response Time Distribution

Response time distribution at workload 8,000

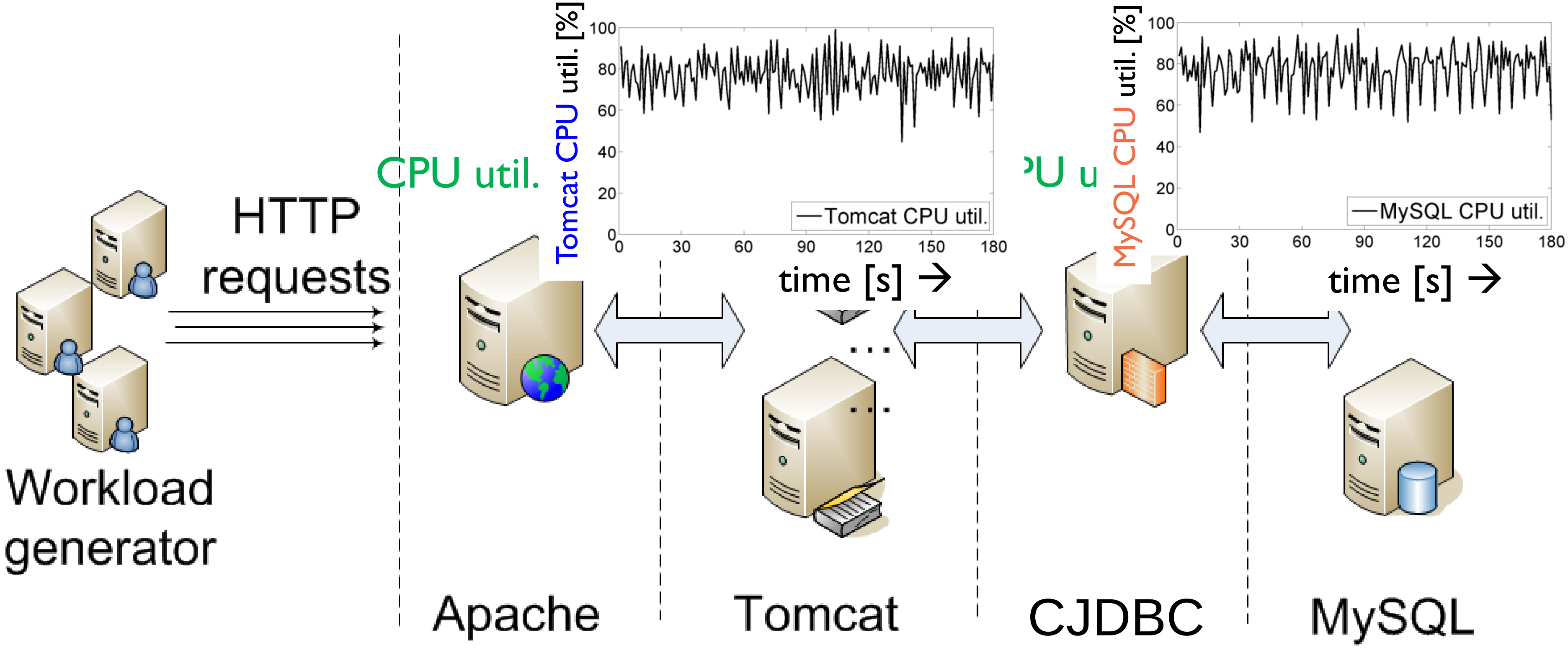
Log scale



The percentage of requests over 2 seconds is more than 5%

No Resources Are Saturated

- Workload is CPU intensive
  - Disk I/O utilization (<5%), network I/O utilization (< 20%), Memory usage (<40%);



Transient Bottlenecks: Sources and Detection

- Sources: We find that other than bursty workload, system environmental conditions:
  - Dynamic Voltage and Frequency Scaling (DVFS)
  - JVM garbage collection

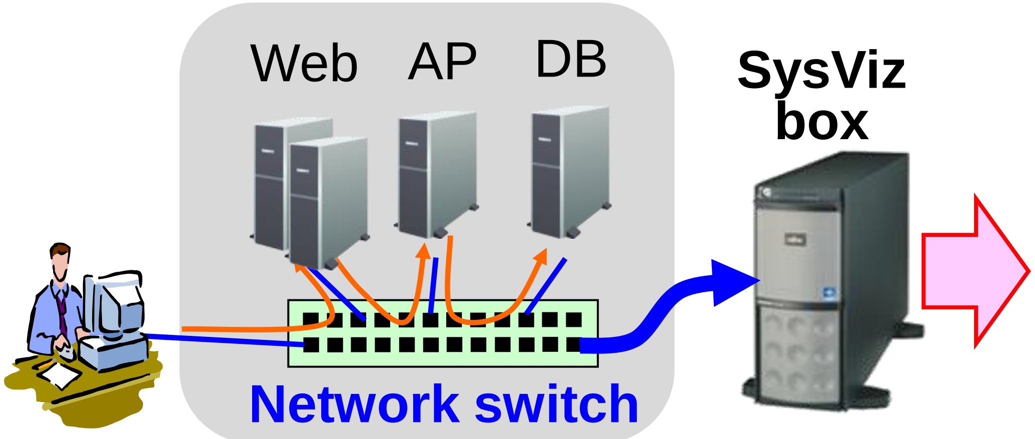
- Detection and Visualization: We develop a fine-grained monitoring method based on passive network tracing in network switches.
  - Negligible monitoring overhead for running applications

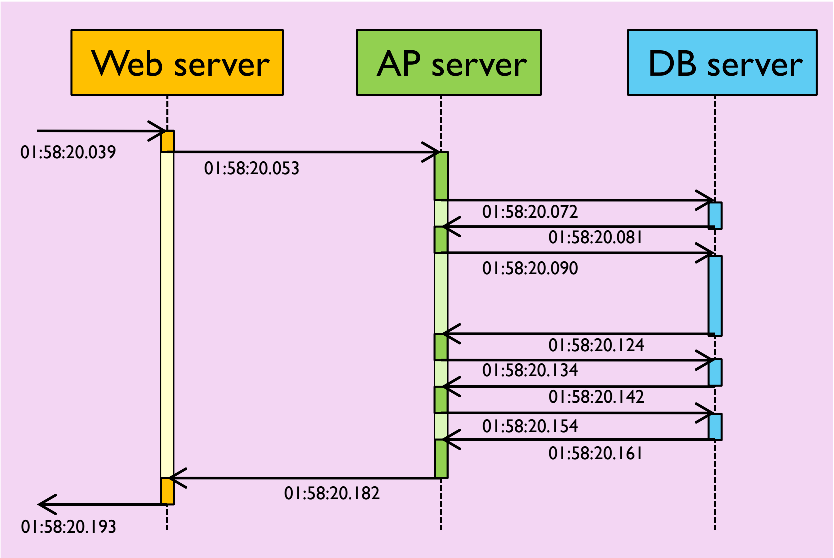
Outline

- □ Introduction & Motivation
- ➡ □ Fine-grained load/throughput analysis method
  1. Data collection via a passive network tracing tool
  2. Calculation of active-load and throughput
  3. Correlation analysis
- □ Two Case Studies
  - ◆ Dynamic Voltage and Frequency Scaling (DVFS)
  - ◆ JVM garbage collection
- □ Conclusion & Future Works

Step1: Data Collection through Passive Network Tracing

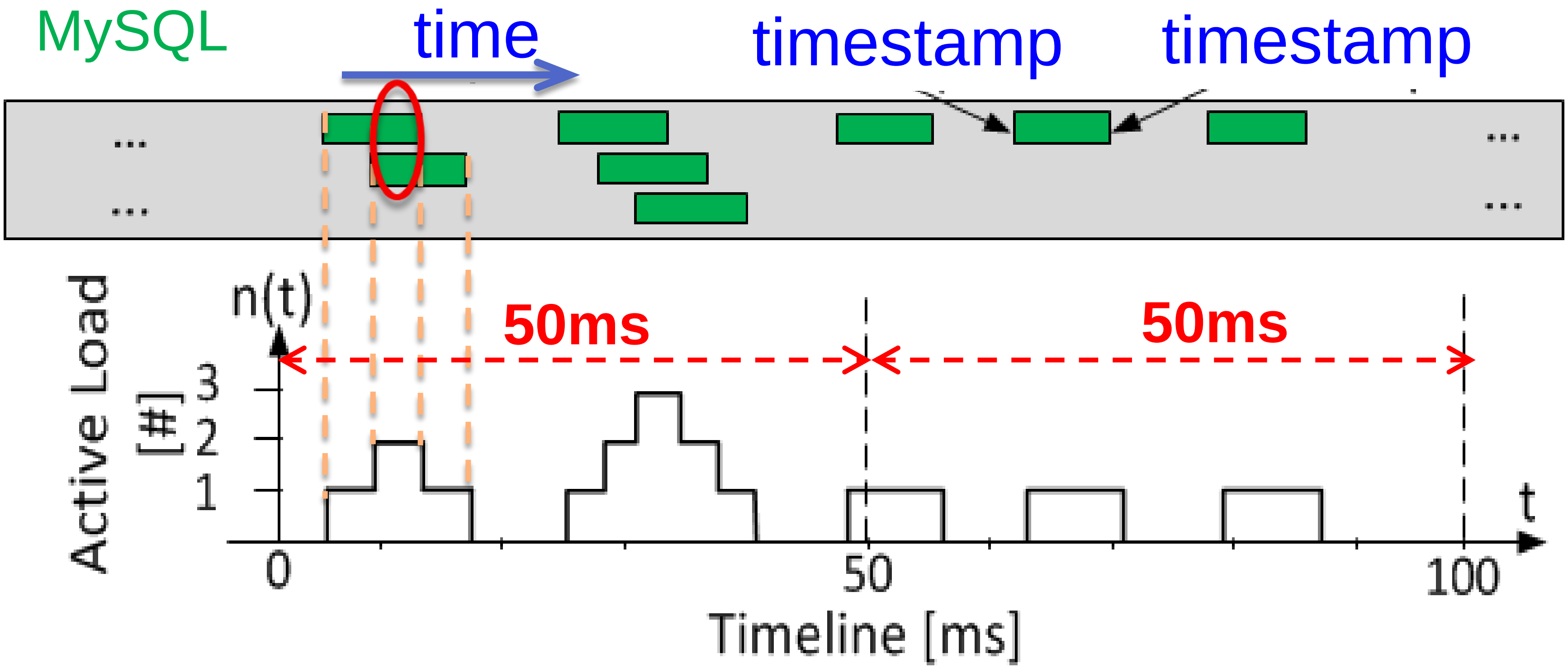
- Collect interaction messages in the system using SysViz to measure fine-grained active load and throughput on each server.
  - **Active load**: The # of concurrent requests in a server
  - **Throughput**: The # of completed requests of a server



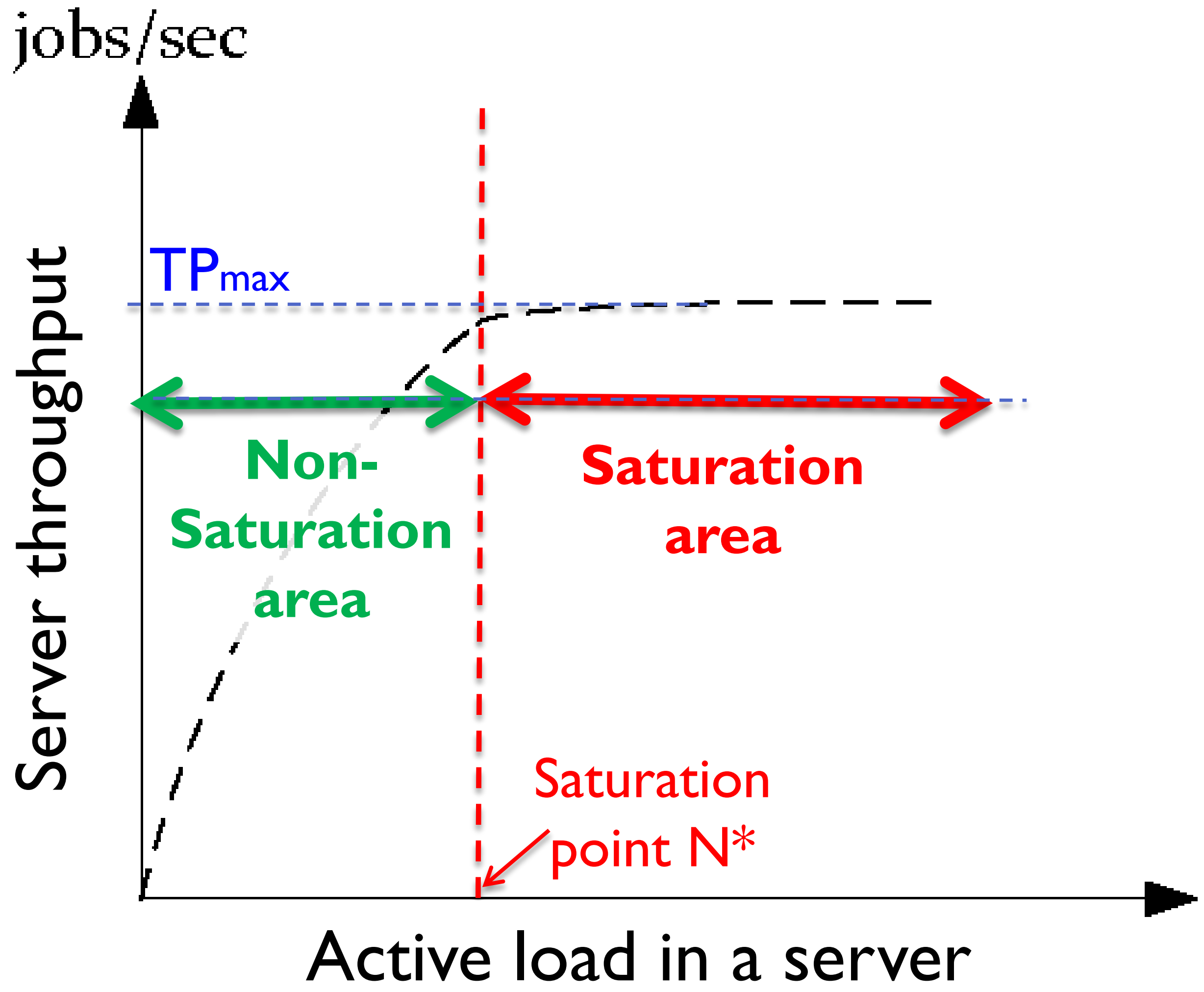


Step2: Fine-Grained Active Load Calculation in a Server

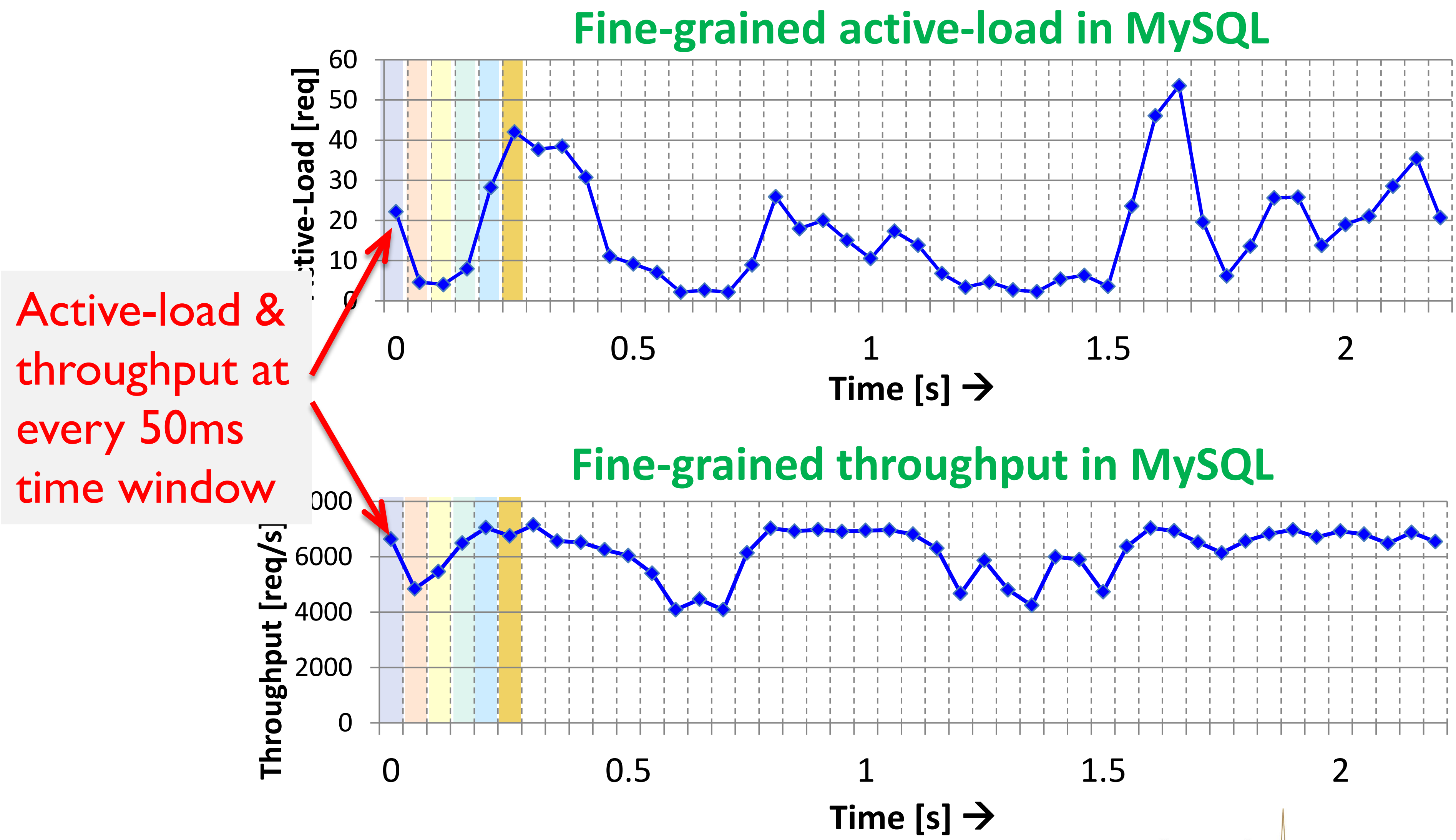
SysViz monitoring for MySQL



Step3: Active-Load/Throughput Correlation Analysis



Step3: Active-Load/Throughput Analysis for MySQL at WL 12,000

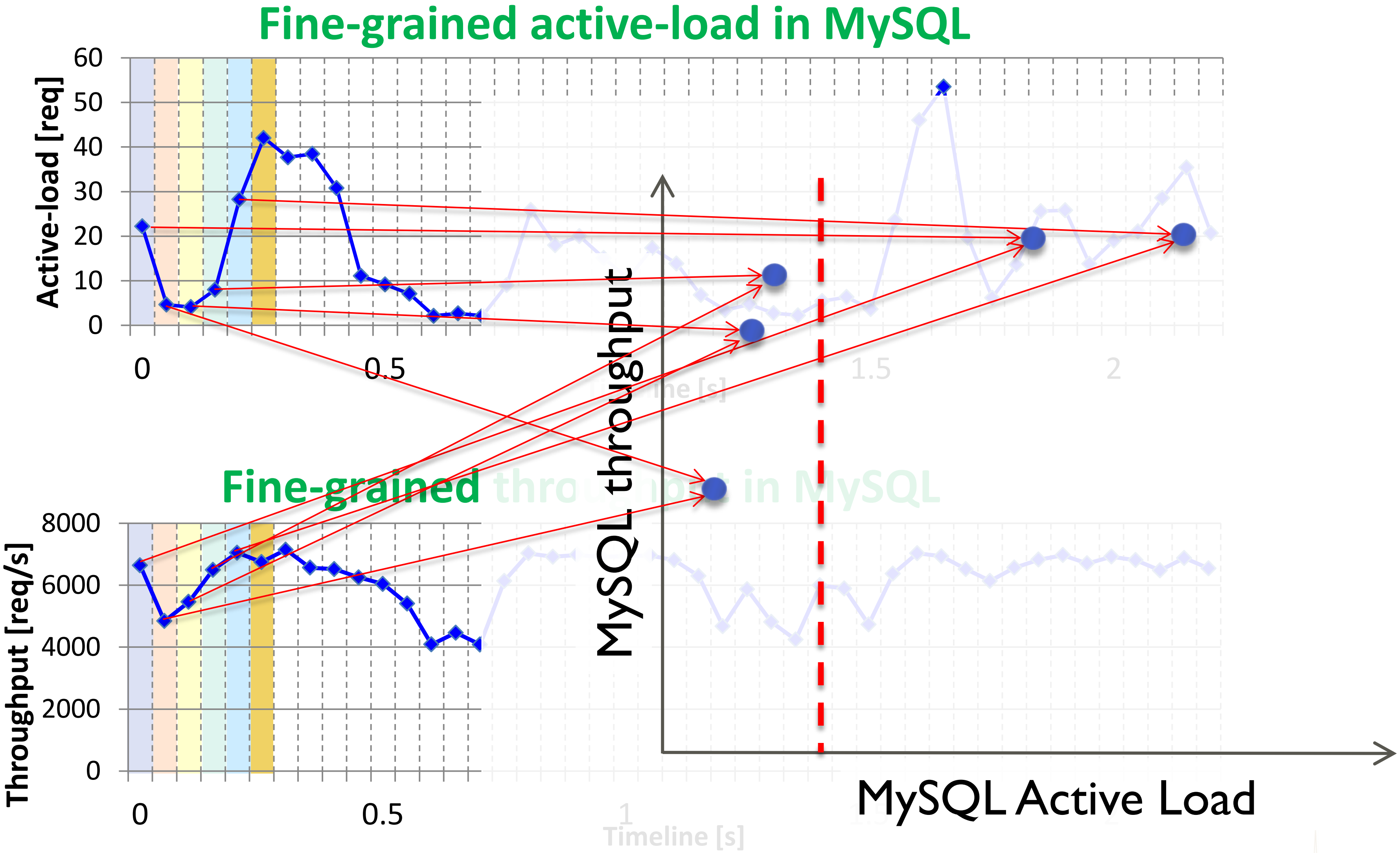


Fine-grained active-load in MySQL

Fine-grained throughput in MySQL

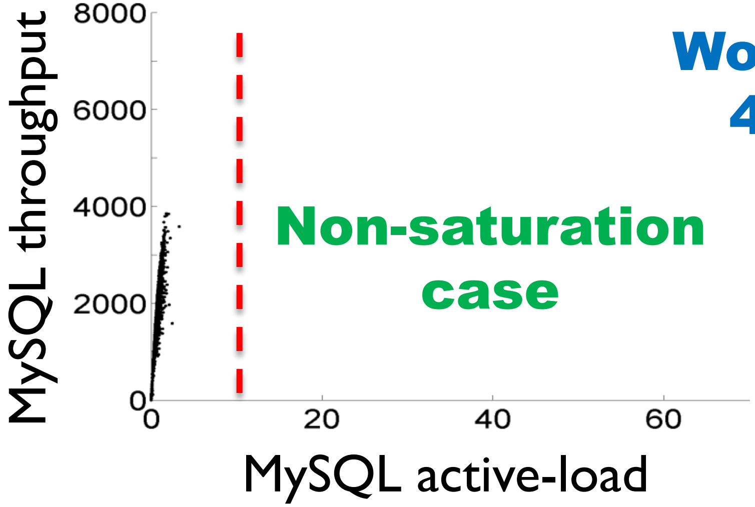
Active-load & throughput at every 50ms time window

Step3: Active-Load/Throughput Analysis for MySQL at WL 12,000 (Cont.)



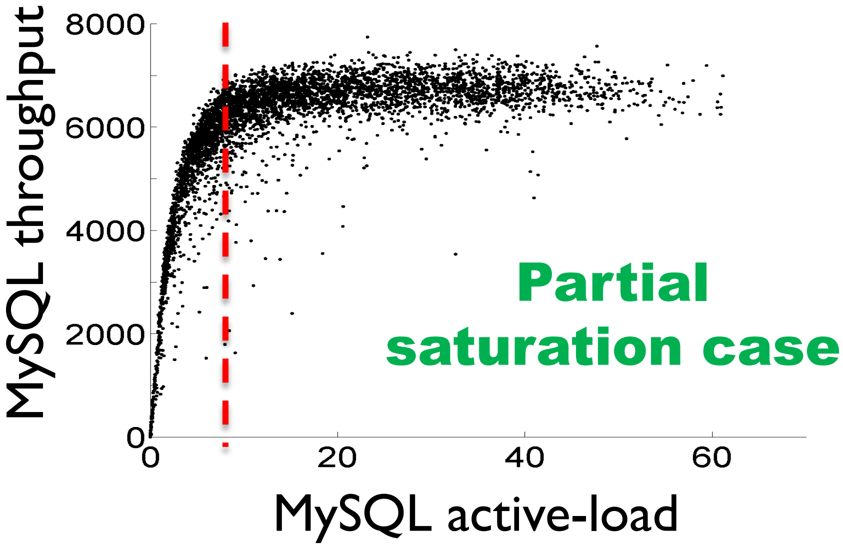
Three Typical Cases of Active-Load/Throughput Analysis

Workload 4,000



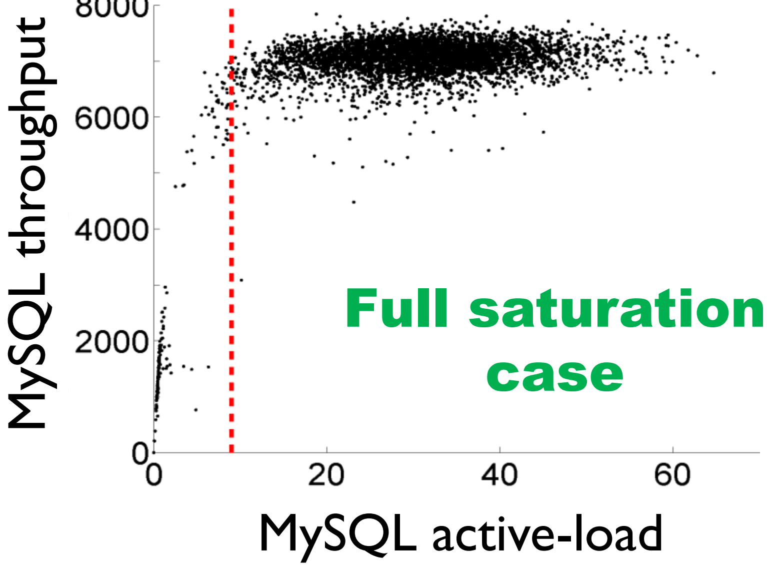
Non-saturation case

Workload 12,000



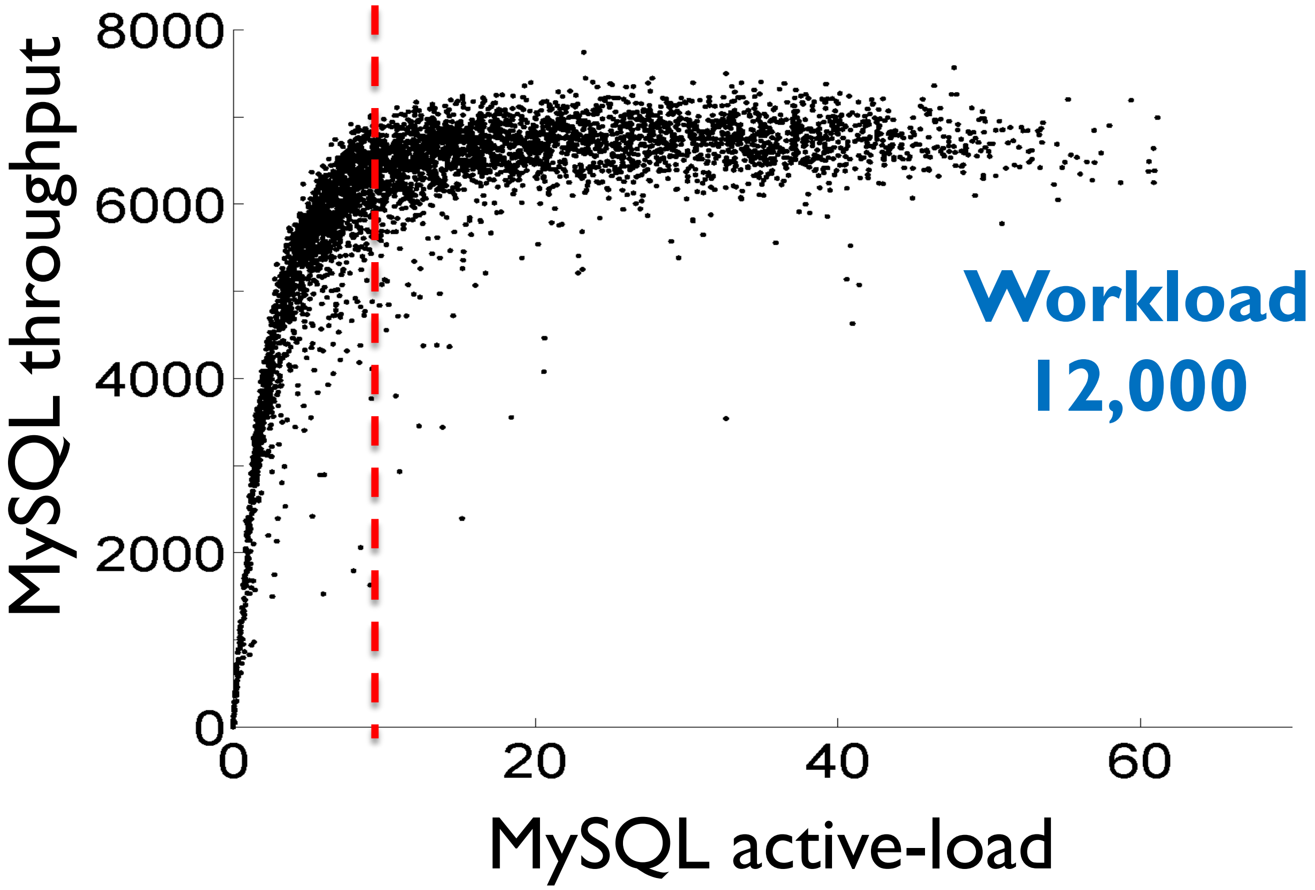
Partial saturation case

Workload 16,000



Full saturation case

Transient Bottlenecks in the Partial Saturation Case



3-D demo

Outline

- Introduction & Motivation
- Fine-grained load/throughput analysis method
  1. Data collection via a passive network tracing tool
  2. Calculation of active-load and throughput
  3. Correlation analysis
- Two Case Studies
  - ➡ Dynamic Voltage and Frequency Scaling (DVFS)
  - JVM garbage collection
- Conclusion & Future Works

Transient Bottlenecks Caused by DVFS

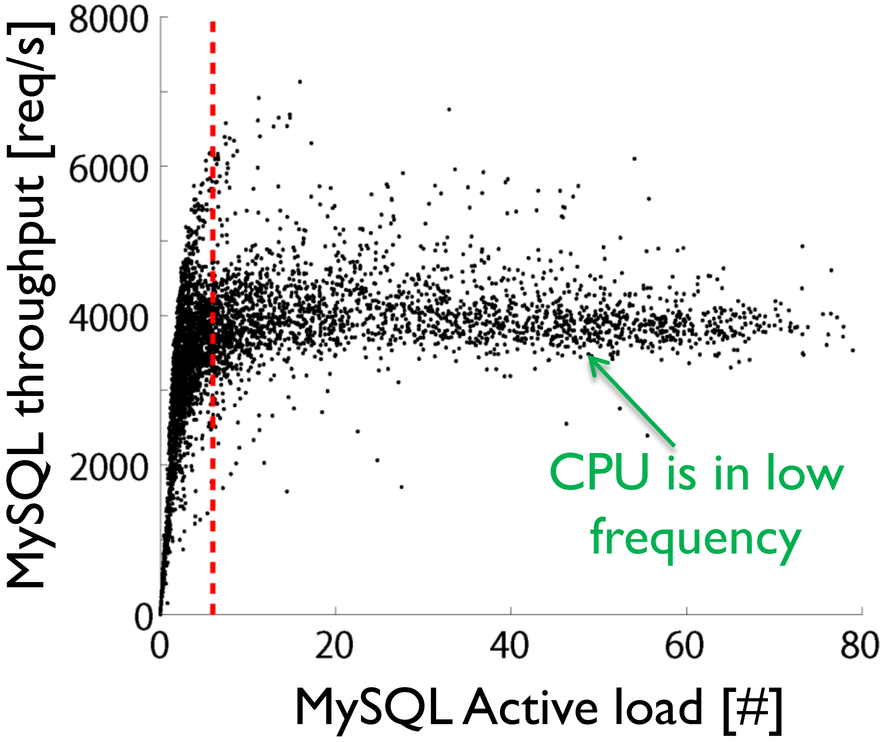
- DVFS is designed to adjust CPU frequency to meet instantaneous performance needs while minimizing power consumption

P-state	P0	P1	P4	P5	P8
CPU Frequency [MHz]	**2261**	2128	1729	1596	**1197**

- We found that the anti-synchrony between DVFS adjustment period and workload burst cycles causes frequent transient bottlenecks.
  - Dell's BIOS-level DVFS control

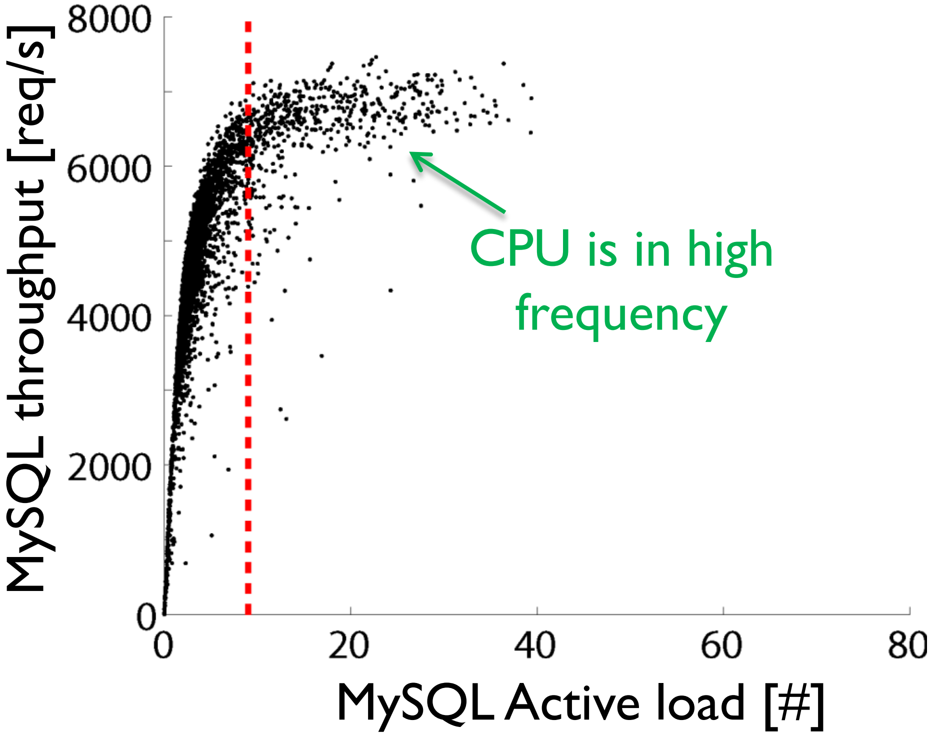
Transient Bottlenecks of MySQL at Workload 8,000

DVFS On case



CPU is in low frequency

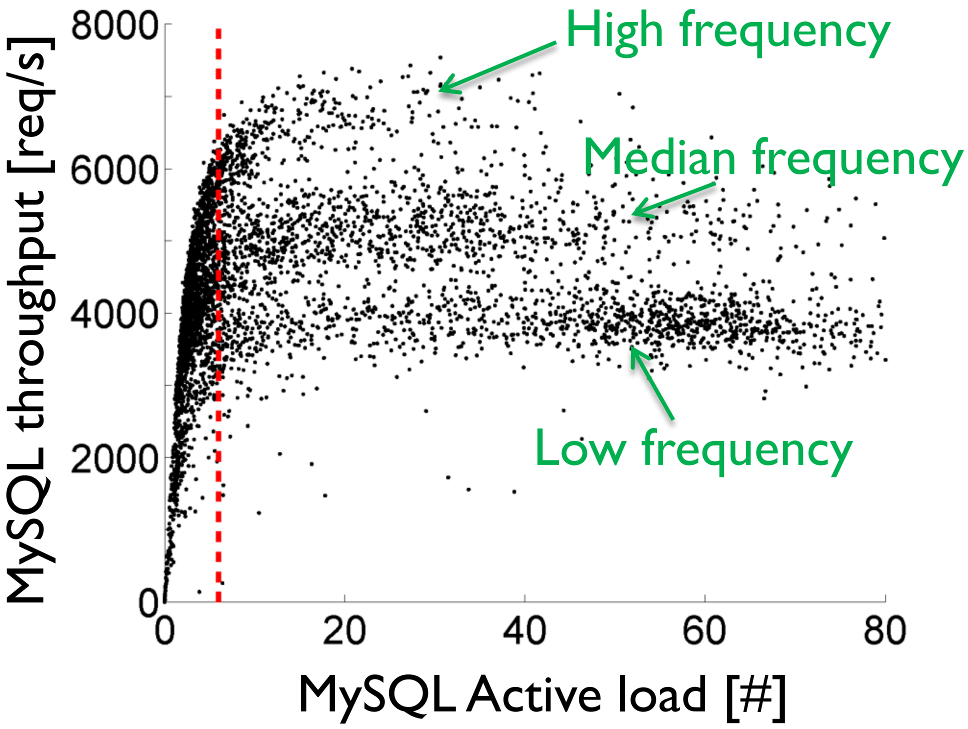
DVFS Off case



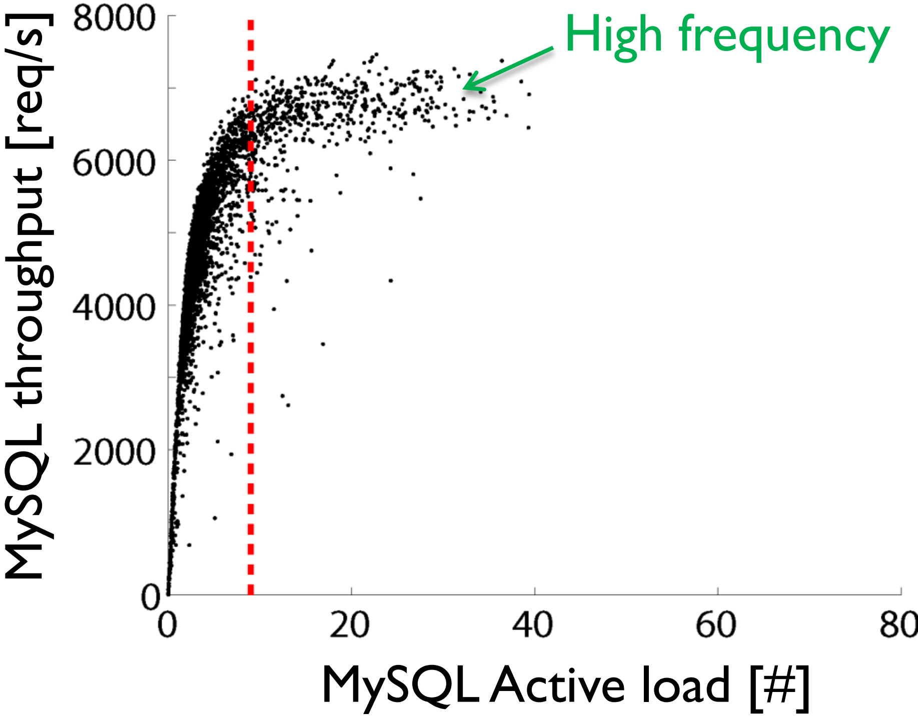
CPU is in high frequency

Transient Bottlenecks of MySQL at Workload 10,000

DVFS On case

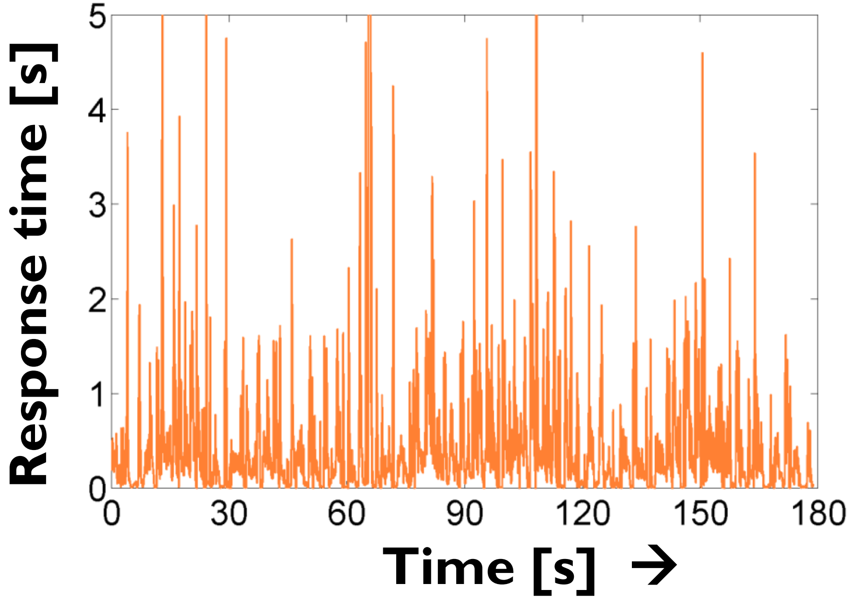


DVFS Off case

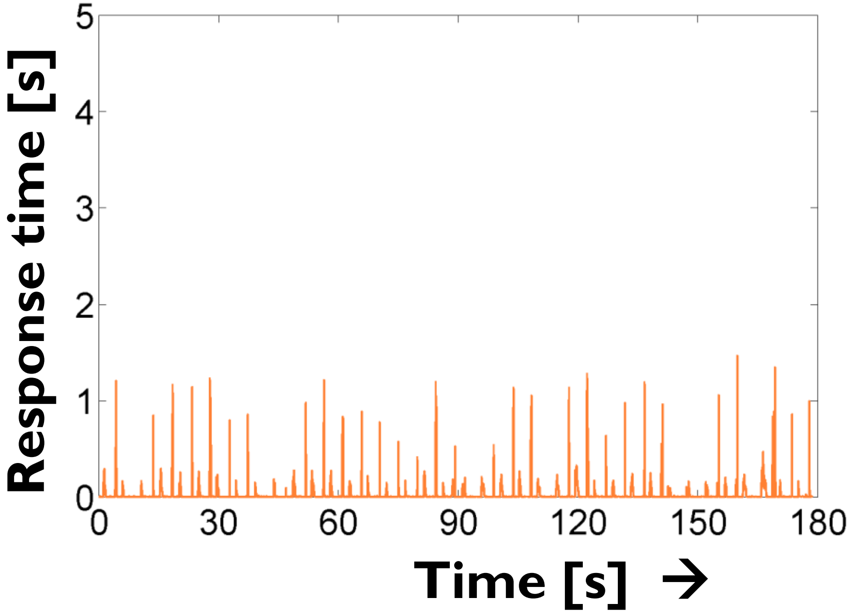


System Response Time with DVFS On/Off at Workload 10,000

DVFS On case



DVFS Off case



Outline

- □ Introduction & Motivation
- □ Fine-grained load/throughput analysis method
  1. Data collection via a passive network tracing tool
  2. Calculation of active-load and throughput
  3. Correlation analysis
- □ Two Case Studies
  - ◆ Dynamic Voltage and Frequency Scaling (DVFS)
  - ◆ JVM garbage collection
- ➡ □ Conclusion & Future Works

Conclusion & Future Work

Transient bottlenecks can cause long-tail response time distributions of an n-tier application.

We developed a **fine-grained active-load/throughput analysis method** which can detect and visualize transient bottlenecks.

Ongoing work: more analysis of different types of workloads and more system factors that cause transient bottlenecks.

Thank You. Any Questions?

Qingyang Wang
qywang@cc.gatech.edu

Georgia Tech College of Computing

cercs

FUJITSU