



**Hasso
Plattner
Institut**

IT Systems Engineering | Universität Potsdam

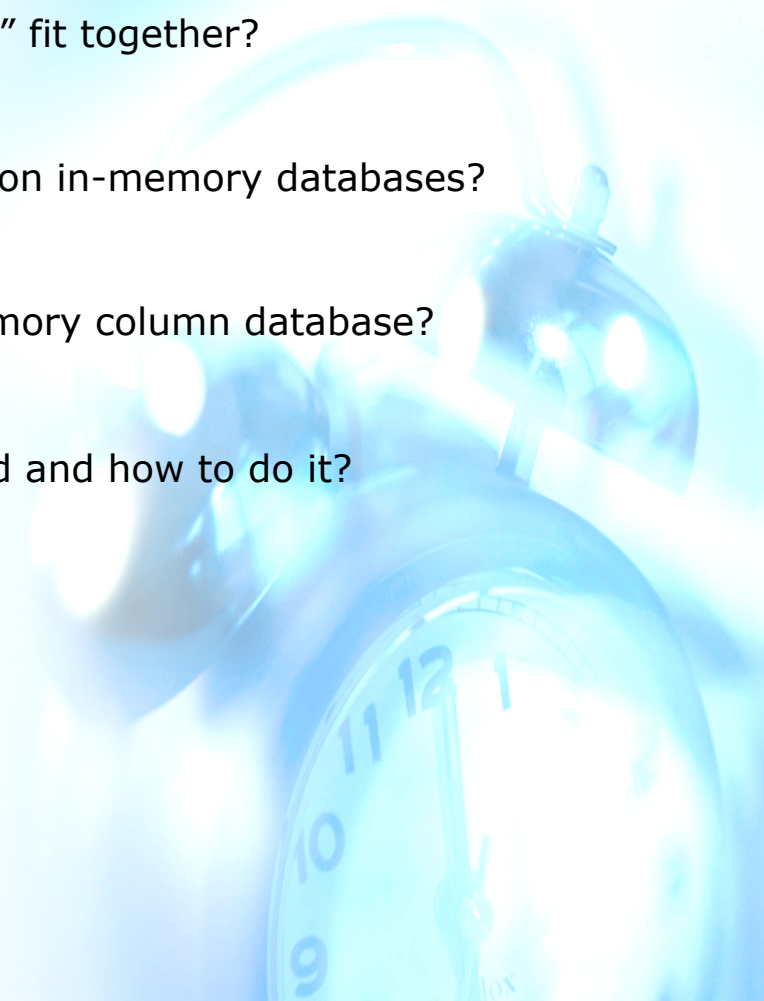
Running In-Memory Databases in the Cloud

***Predicting Response Times and Automating
Cluster Management Tasks***

Jan Schaffner

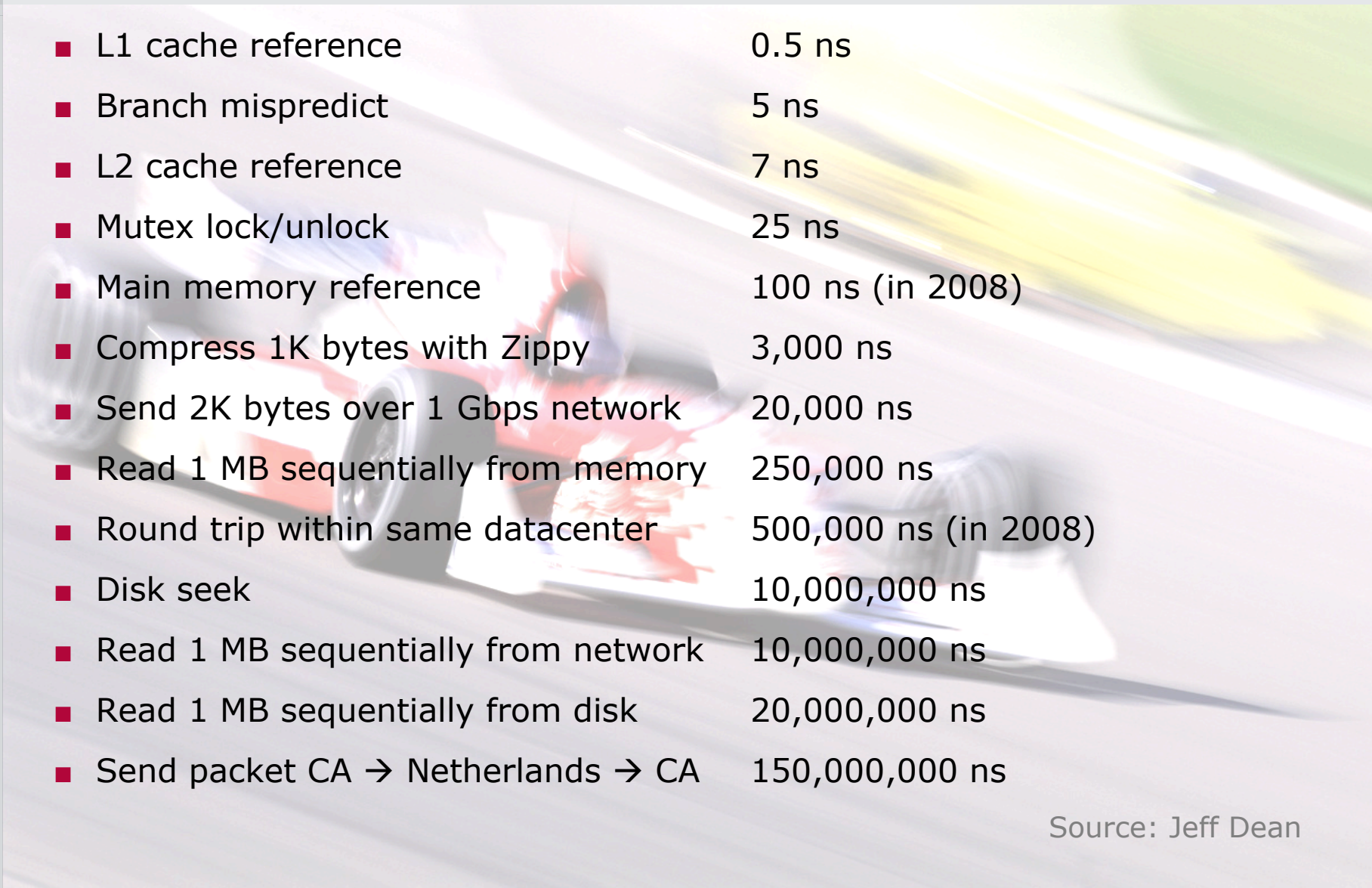
What to take home from this talk?

Answers to four questions:

- How do “in-memory” and “cloud computing” fit together?
 - Does virtualization have a negative impact on in-memory databases?
 - How to predict response times of an in-memory column database?
 - Why should data in the cluster be replicated and how to do it?
- 

**How do “in-memory” and
“cloud computing” fit together?**

Numbers everyone should know

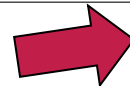


■ L1 cache reference	0.5 ns
■ Branch mispredict	5 ns
■ L2 cache reference	7 ns
■ Mutex lock/unlock	25 ns
■ Main memory reference	100 ns (in 2008)
■ Compress 1K bytes with Zip	3,000 ns
■ Send 2K bytes over 1 Gbps network	20,000 ns
■ Read 1 MB sequentially from memory	250,000 ns
■ Round trip within same datacenter	500,000 ns (in 2008)
■ Disk seek	10,000,000 ns
■ Read 1 MB sequentially from network	10,000,000 ns
■ Read 1 MB sequentially from disk	20,000,000 ns
■ Send packet CA → Netherlands → CA	150,000,000 ns

Source: Jeff Dean

Some more recent numbers...

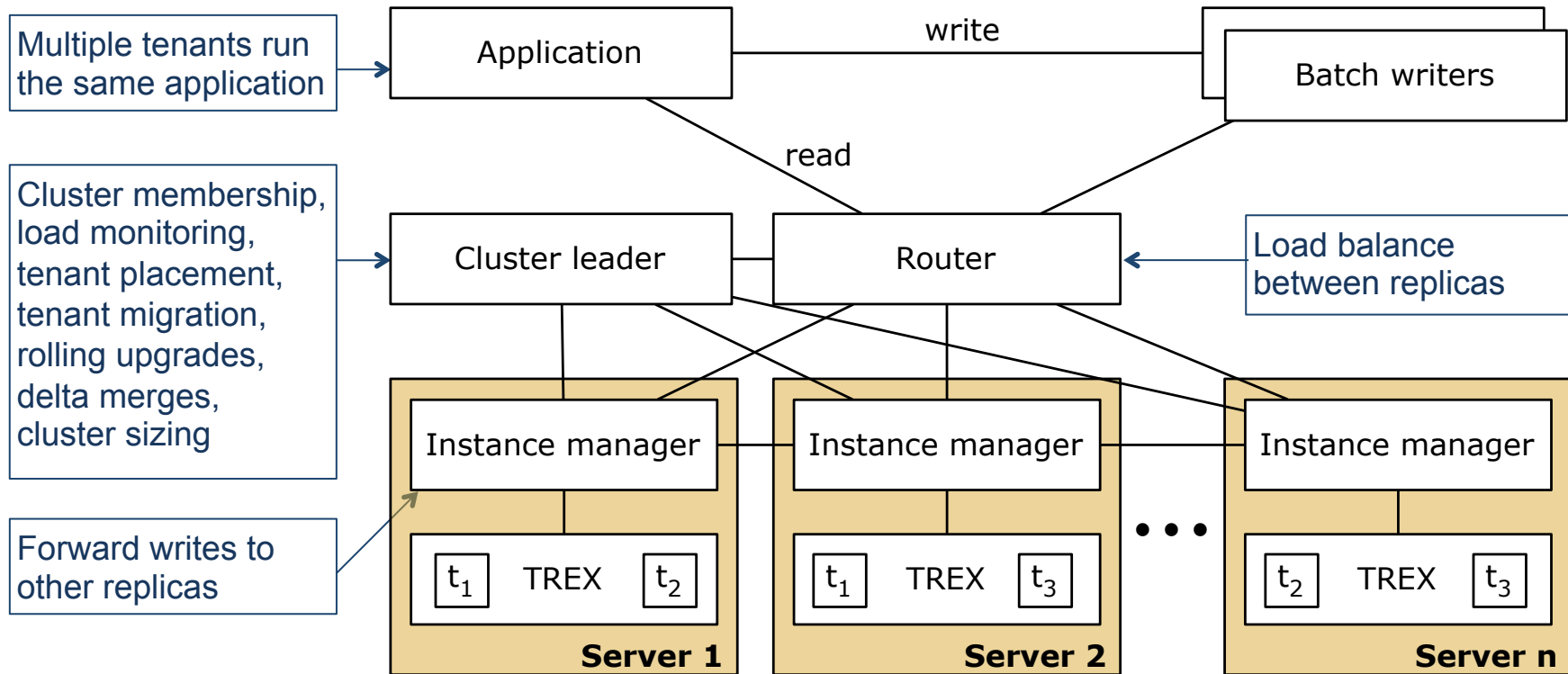
	Type	Device / Medium	Latency	Throughput
Same core	Storage	L1 cache read (local)	1.3 ns	364.8 Gbps
	Storage	L2 cache read (local)	3.4 ns	248.8 Gbps
	Storage	L3 cache read (local)	13 ns	209.6 Gbps
Same die	Storage	L1 cache read (remote, same die)	13 - 28.3 ns	75.2 - 154.4 Gbps
	Storage	L2 cache read (remote, same die)	13 - 25.5 ns	105.6 - 157.6 Gbps
	Storage	L3 cache read (remote, same die)	13 - 22.2 ns	157.6 - 209.6 Gbps
Same board	Storage	L1 cache read (remote, via QPI)	58 - 109 ns	44.8 - 72 Gbps
	Storage	L2 cache read (remote, via QPI)	58 - 109 ns	44.8 - 73.6 Gbps
	Storage	L3 cache read (remote, via QPI)	58 - 109 ns	44.8 - 73.6 Gbps
	Storage	DRAM (Nehalem)	65 - 106 ns	160 - 204.8 Gbps / socket
Same machine	Interconnect	SATA 3.0	at least 1 μ s	6 Gbps
	Interconnect	Serial Attached SCSI	at least 1 μ s	6 Gbps
	Interconnect	PCI Express	3.8 - 5 μ s	4 Gbps x number of lanes
	Storage	Magnetical disk read / write	3.2 - 13 ms	0.96 - 1.12 Gbps
	Storage	Solid State Disk read	65 μ s	1.9 Gbps
Network	Interconnect	RDMA over InfiniBand	1 - 3 μ s	2.5 - 10 Gbps x number of channels
	Interconnect	RDMA over iWARP	6 μ s	10 Gbps / link
	Interconnect	10Gb Ethernet	20 μ s	10 Gbps / link
	Interconnect	Fibre channel	3 - 10 μ s (add 1 ms per 100 km)	8 Gbps / channel



Main memory should be the system of record

- Bandwidth:
 - Disk: 120 MB/s/controller
 - DRAM (x86 + FSB): 10.4 GB/s/board
 - DRAM (Nehalem): 25.6 GB/s/socket
- Latency:
 - Disk: 3.2 - 13 milliseconds
 - InfiniBand: 1 - 3 microseconds
 - DRAM: 65 - 105 nanoseconds
- **Claim:** Two machines + network is better than one machine + disk
 - Log to disk on a single node:
> 3,200 μ s (best case)
 - Transactions only in memory but on two nodes:
< 25 μ s (worst case using 10Gbit Ethernet)

Rock cluster architecture (active / active configuration)



Design choices for a cloud analytics database

- Multi-master replication
 - Two copies of the data
 - Load balancing both reads and (monotonic) writes
 - Eventual consistency via simple write propagation and MVCC (nodes might lag behind slightly)

- High-end hardware
 - Nehalem for high memory bandwidth (more tenants per node)
 - Fast interconnects to minimize write lag

- Virtualization
 - Ease of deployment / administration
 - Consolidation / multi-tenancy

Second question

**Does virtualization have a negative impact
on in-memory databases?**

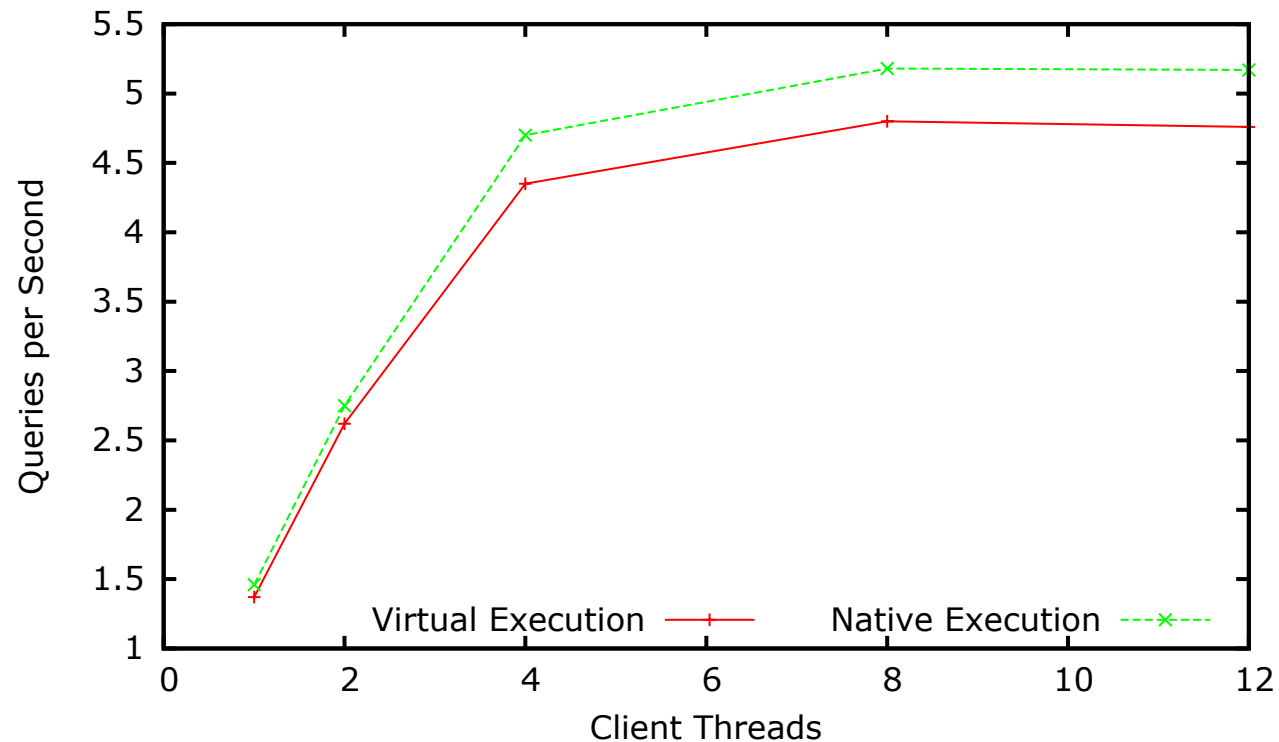
Why virtualization?

- In-memory column databases are ideal for processing mixed OLTP and OLAP workloads
- **But:** In a SaaS environment it seems costly to give every tenant their private database server

- How much consolidation is possible?
 - 3 years worth of sales records from a Fortune 500 retail company: 360 million records and less than 3 GB in memory
 - Typical SaaS customer is an order of magnitude smaller
 - Next door we have a machine with 2TB of main memory

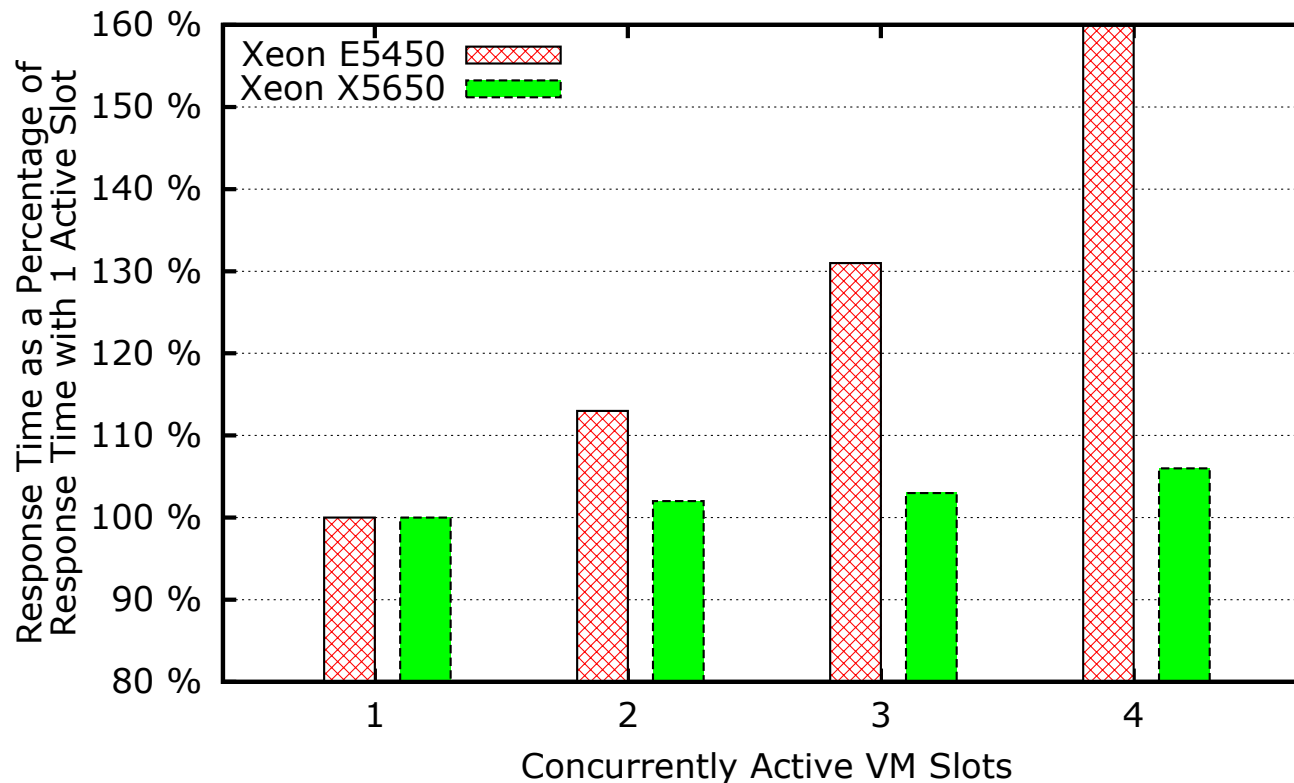
Impact of virtualization

- Run multi-tenant OLAP benchmark on either:
 - one TREX instance directly on the physical host vs.
 - one TREX instance inside VM on the physical host
- Overhead is approximately 7% (both in response time and throughput)



Impact of virtualization (contd.)

- Virtualization is often used to get “better” system utilization
 - What happens when a physical machine is split into multiple VMs?
 - Burning CPU cycles does not hurt → memory bandwidth is the limiting factor



Third question

**How to predict response times
of an in-memory column database?**

...in an on-demand setting

Challenges for the provider

- Aim for high amount of consolidation (multi-tenancy)

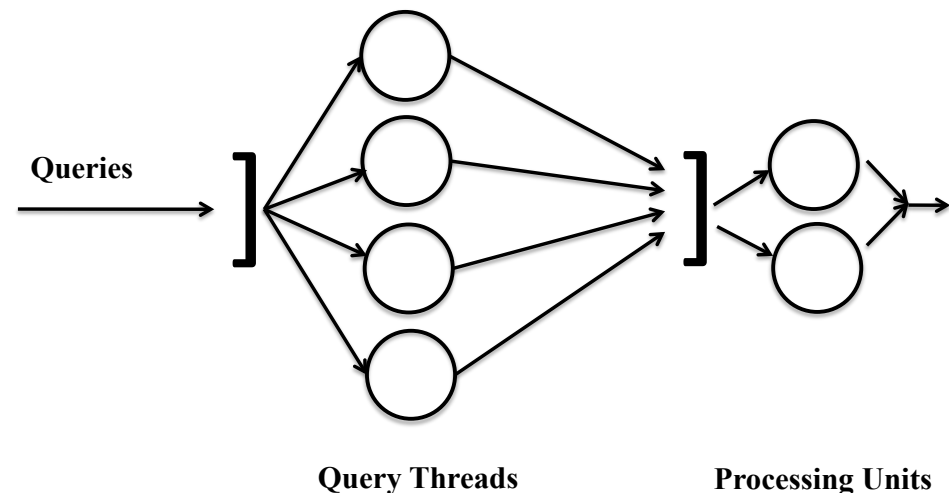
- In a SaaS environment, the question is “how do I guarantee a fixed (low) response time as cheap as possible?”
 - Look at throughput
 - Look at quartiles (e.g. 99th percentile)

- Formulation of desired performance guarantee:
 - Response time goal “1 second in the 99th percentile”
 - Average response time around 200 ms
 - Less than 1% of all queries exceed 1,000 ms
 - Results in a maximum number of concurrent x queries before response time goal is violated

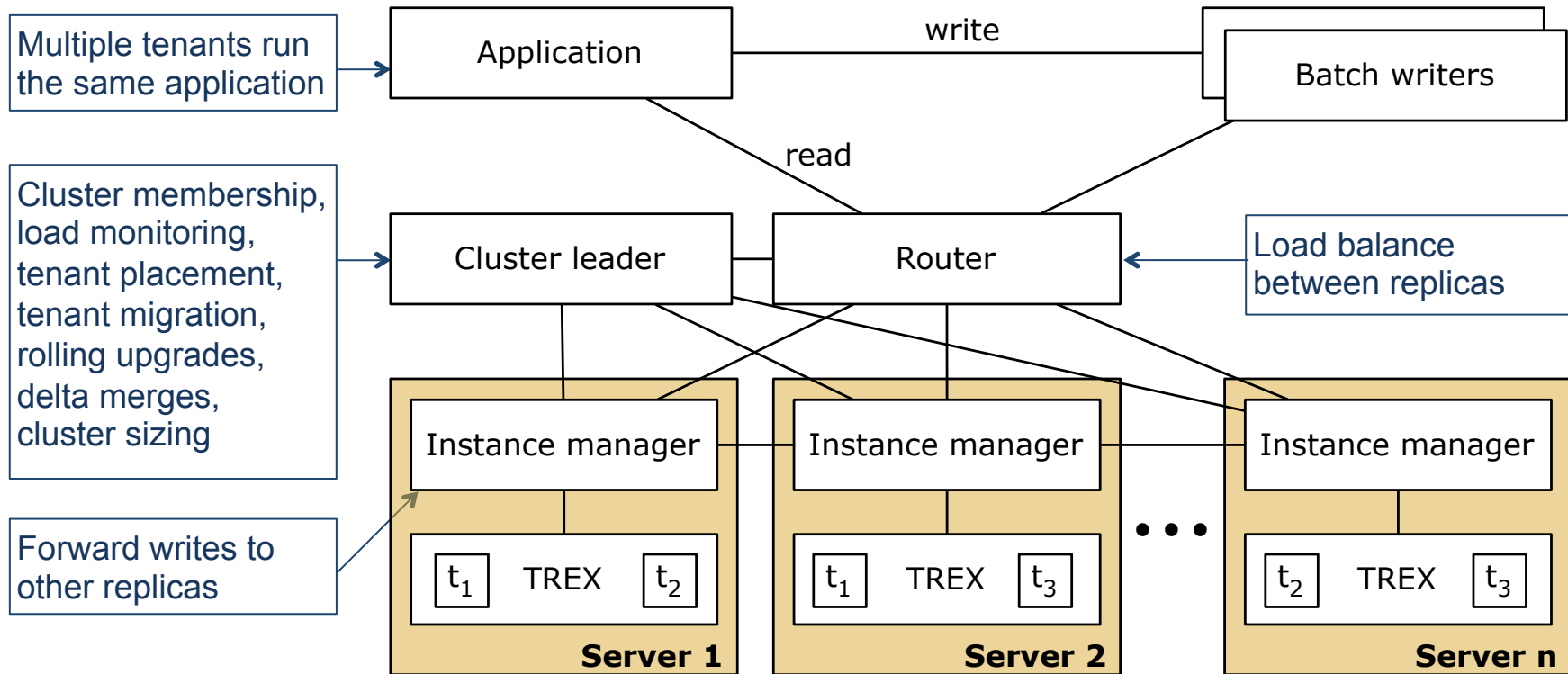


Problem statement

- Given a cluster of in-memory column databases:
 - How many tenants at what request rates can individual servers handle without violation the response time goal?
 - How to estimate the combined load incurred by multiple tenants on a server?
 - How do cluster management operations affect this estimation?
- How to obtain such load estimations?
 - Analytical model
 - Empirical model



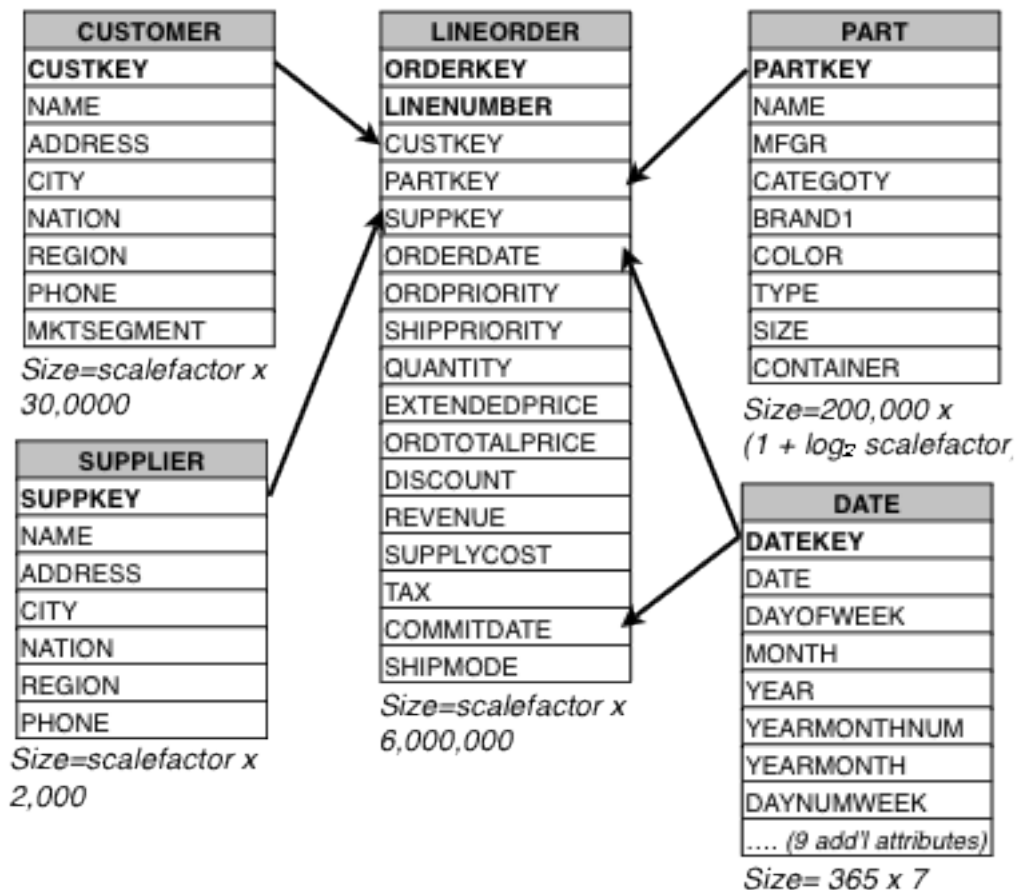
Rock cluster architecture (active / active configuration)



Multi-tenant OLAP benchmark

- Based on SSB¹, a modified version of TPC-H
- One instance of the SSB tables per tenant (*private tables*)

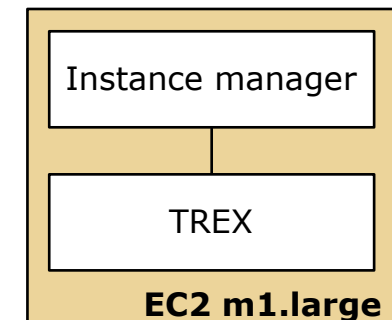
- Added parallel user threads and think time
- Added writes
- Queries grouped into *flights*
- Snapshot isolation
 - scale # of users
 - stay within fixed response time goal



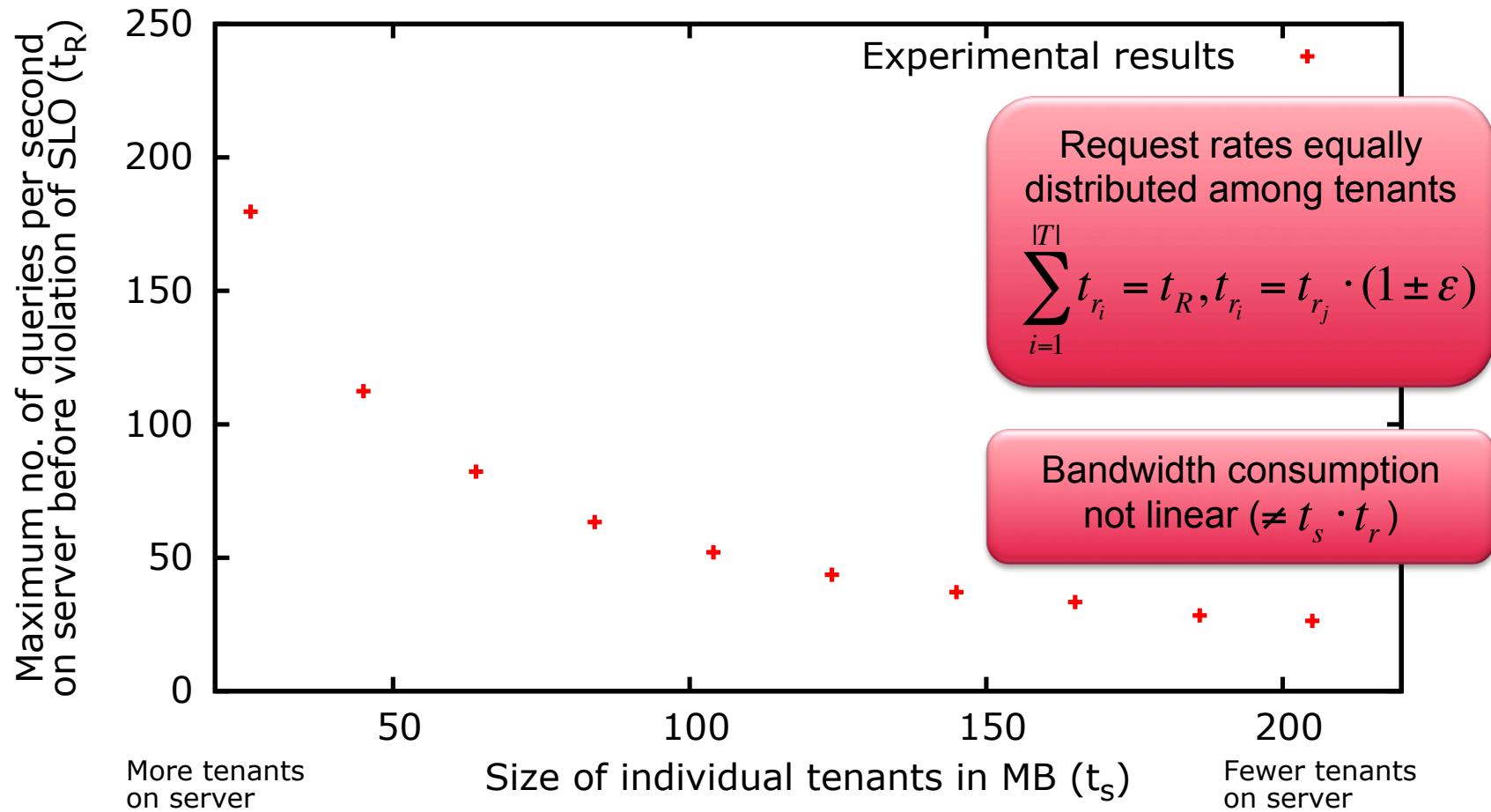
[1] P.E. O'Neil, E.J. O'Neil, X. Chen, S. Revilak. The Star Schema Benchmark and Augmented Fact Table Indexing. In: Performance Evaluation and Benchmarking, TPCTC 2009, p. 237-252, 2009.

Experimental setup

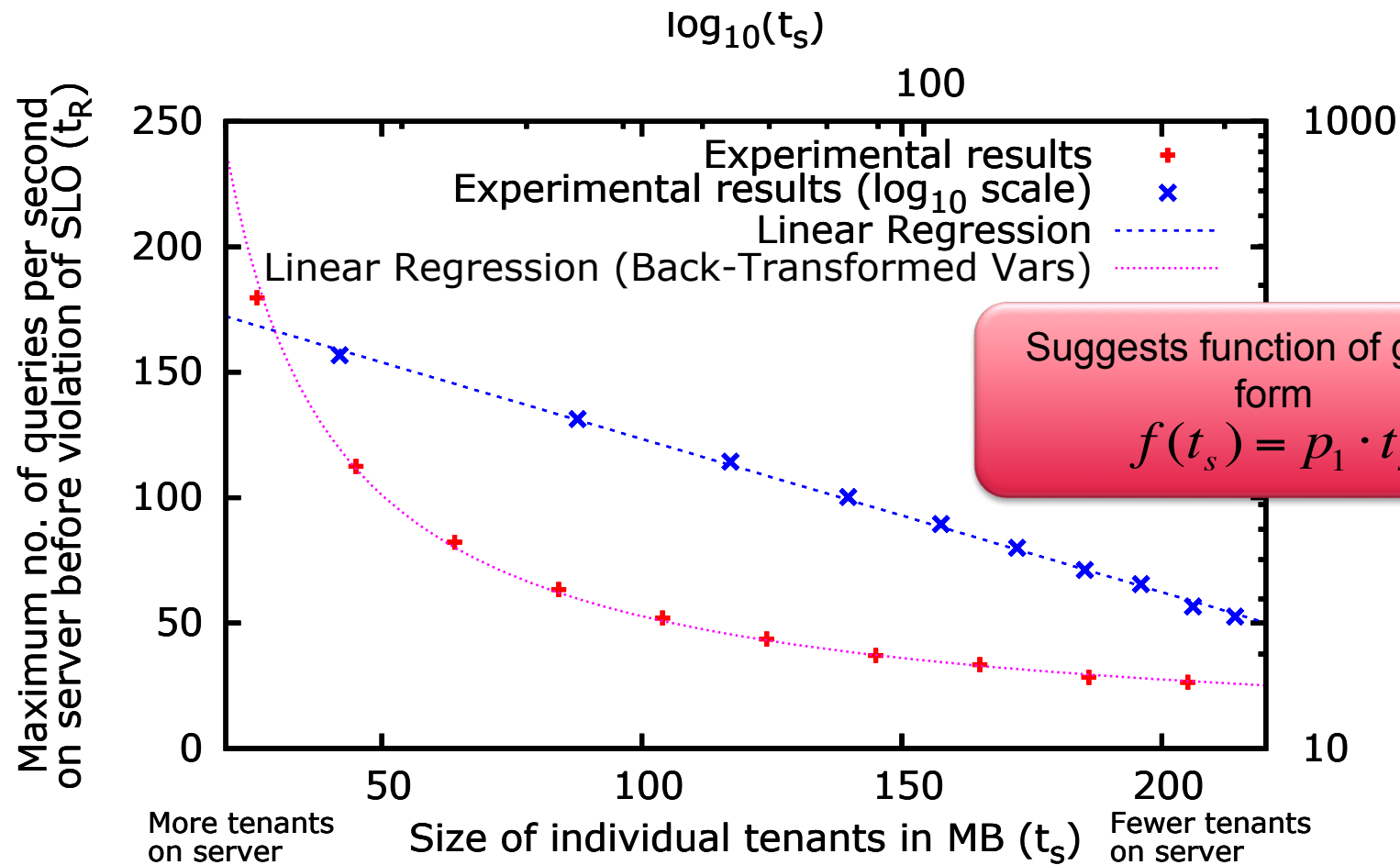
- TREX nodes packaged as EC2 *m1.large* instances
 - 7.5 GB RAM
 - 2 virtual cores
- Routers and cluster leader packaged as EC2 *c1.xlarge* instances
 - 7 GB RAM, 8 virtual cores
 - High I/O performance
- Benchmarking only *a single node*
 - Filling up 40% of the available memory with tenants $t \in T$
 - In each run, all tenants have the same size t_s
 - Tenant size t_s is varied across runs
 - Fixed response time goal independent of t_s (across whole server)
 - All tenants have same base request rate t_r
 - Request rate is increased until server violates SLO



Maximum Request Rate without violation of SLO

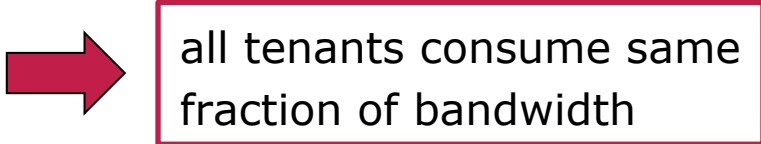


Maximum Request Rate without violation of SLO



...after log-log transformation:
 $\log f(t_s) = \log p_1 + p_2 \cdot \log t_s$

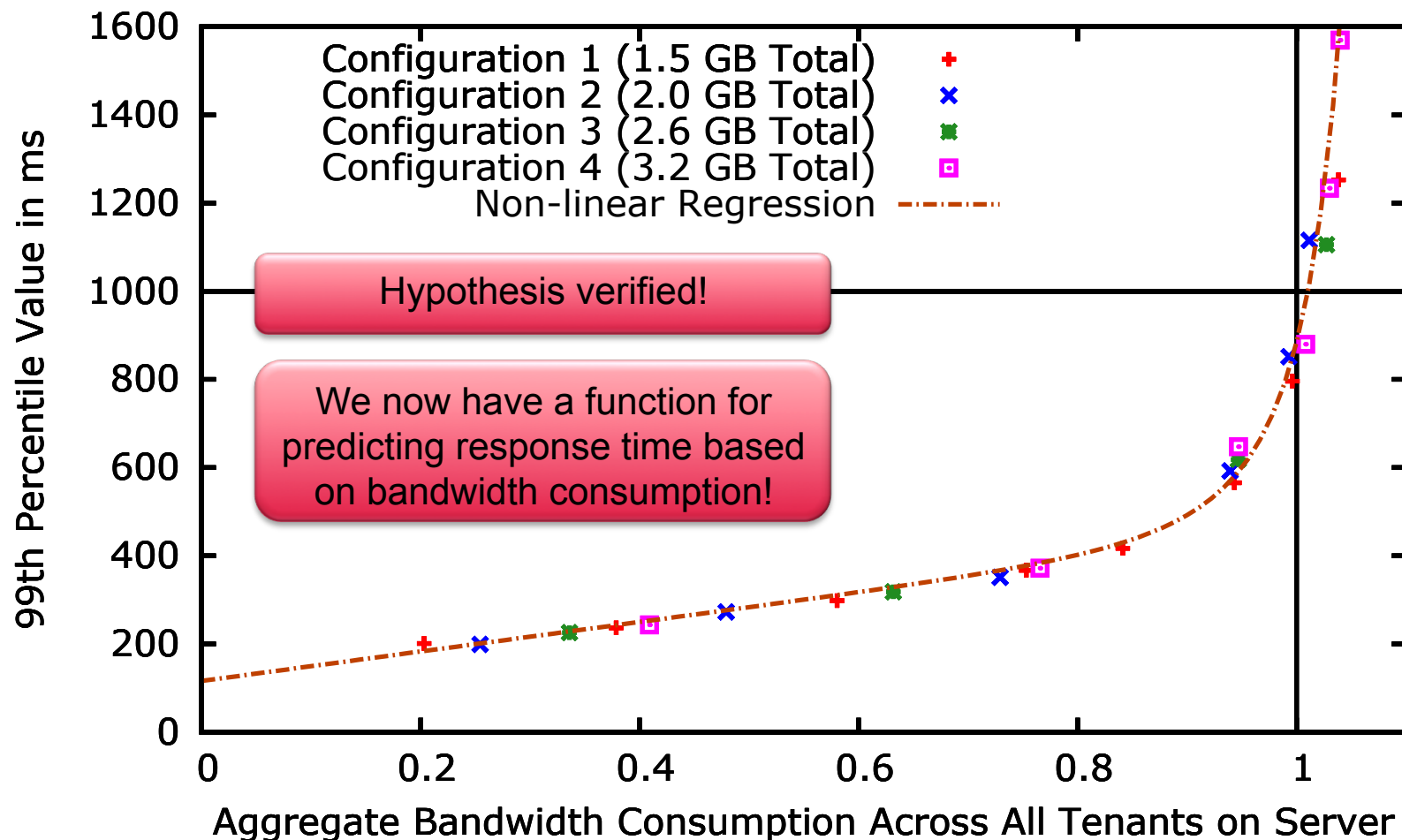
Calculating *bandwidth* consumed by a tenant

- Using linear regression we obtained a function for computing the maximum no. of requests per server without SLO violation, given:
 - all tenants have the same size
 - request rate is equally distributed

all tenants consume same fraction of bandwidth
- **Hypothesis:** SLO violations on a server can also be predicted for sets of tenants of arbitrary sizes and request rates
- We need a metric for quantifying bandwidth consumption per tenant
 - Idea: use function for maximum request rate
 - Make t_r an independent variable
 - Monotonically increasing for increasing t_s, t_r
 - $b(t_s, t_r) \rightarrow [0..1]$

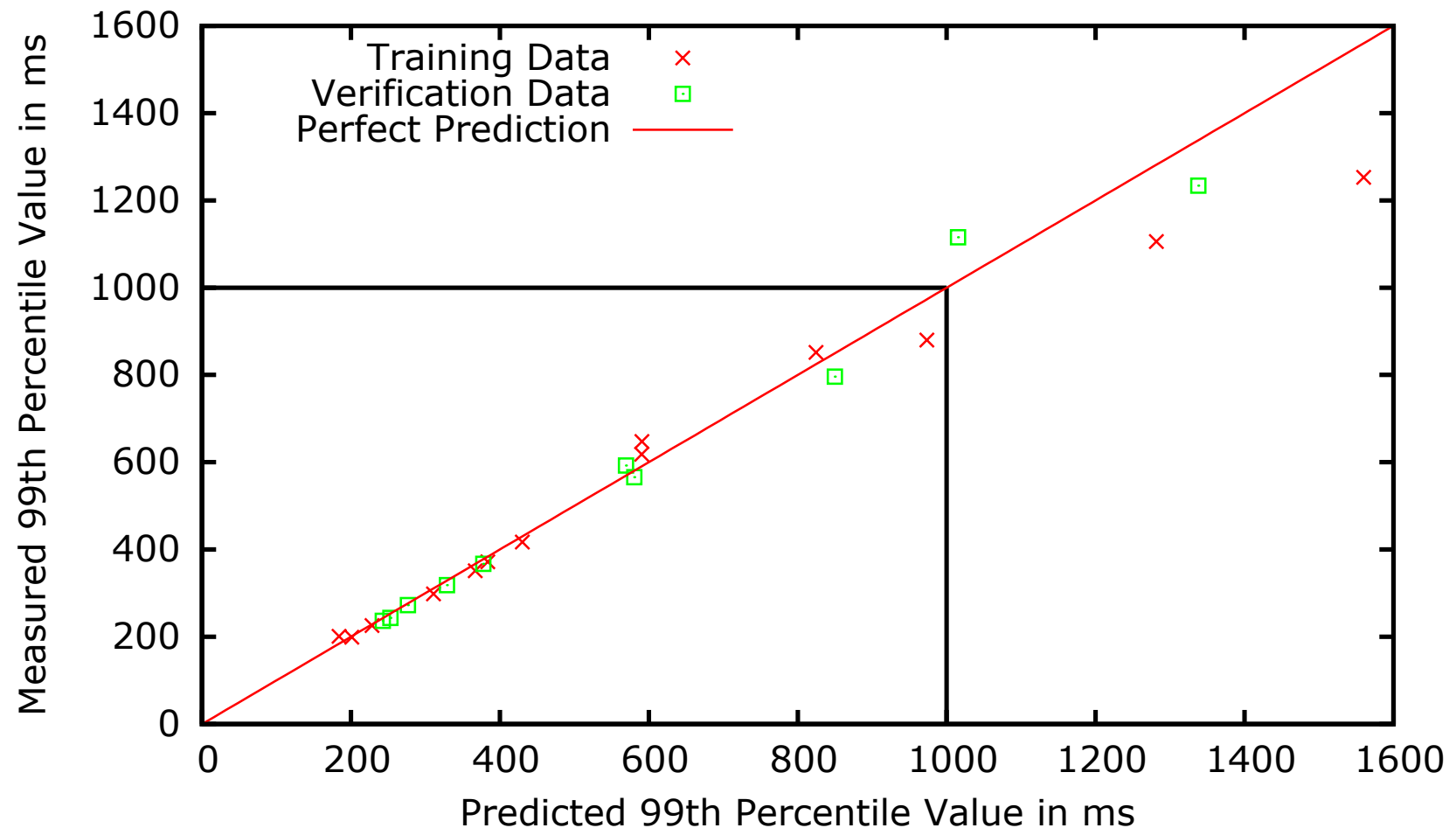
Relation of aggregate *bandwidth* and 99-th percentile value

- Validate our hypothesis that SLO violation can be predicted with differently sized tenants and request rate



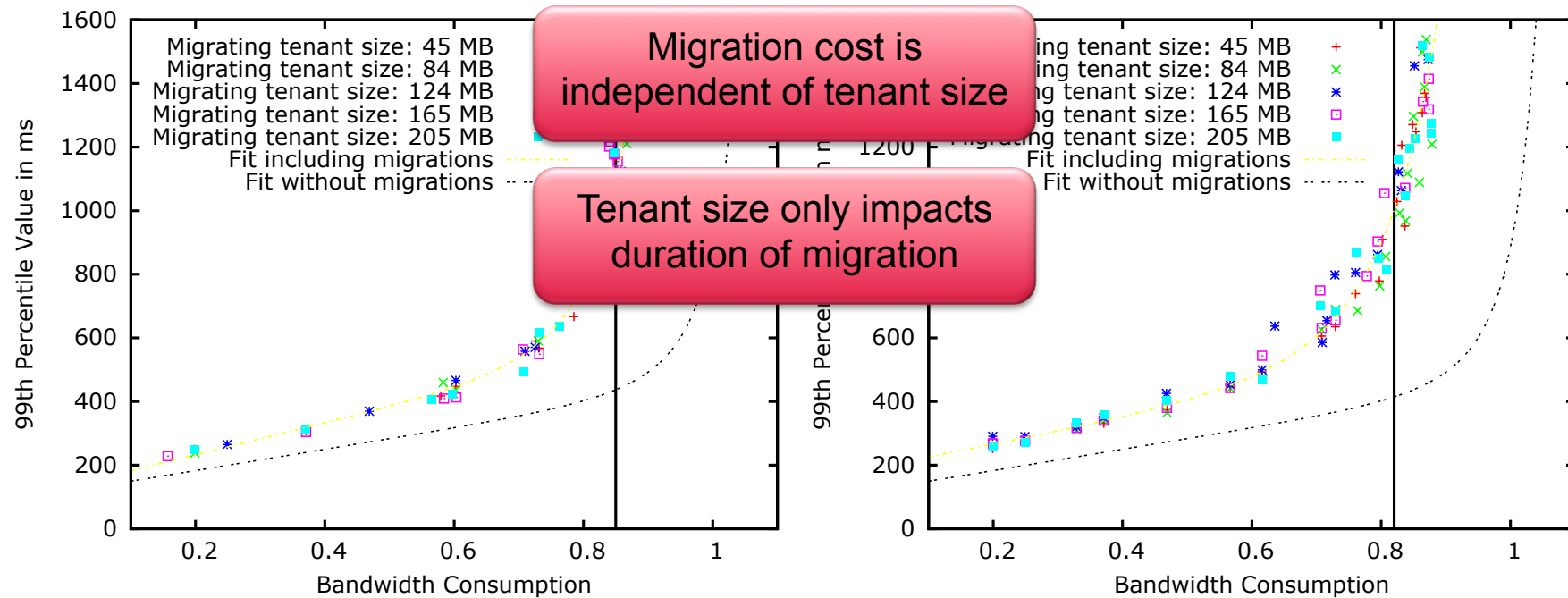
Prediction accuracy

- Split experimental results in training data and test data
- Root mean squared error for predicting values < 1000 ms is 31.21 ms



Extending the model with migrations

- Migration: packing, network transfer, unpacking, preloading into DRAM



- Source node:

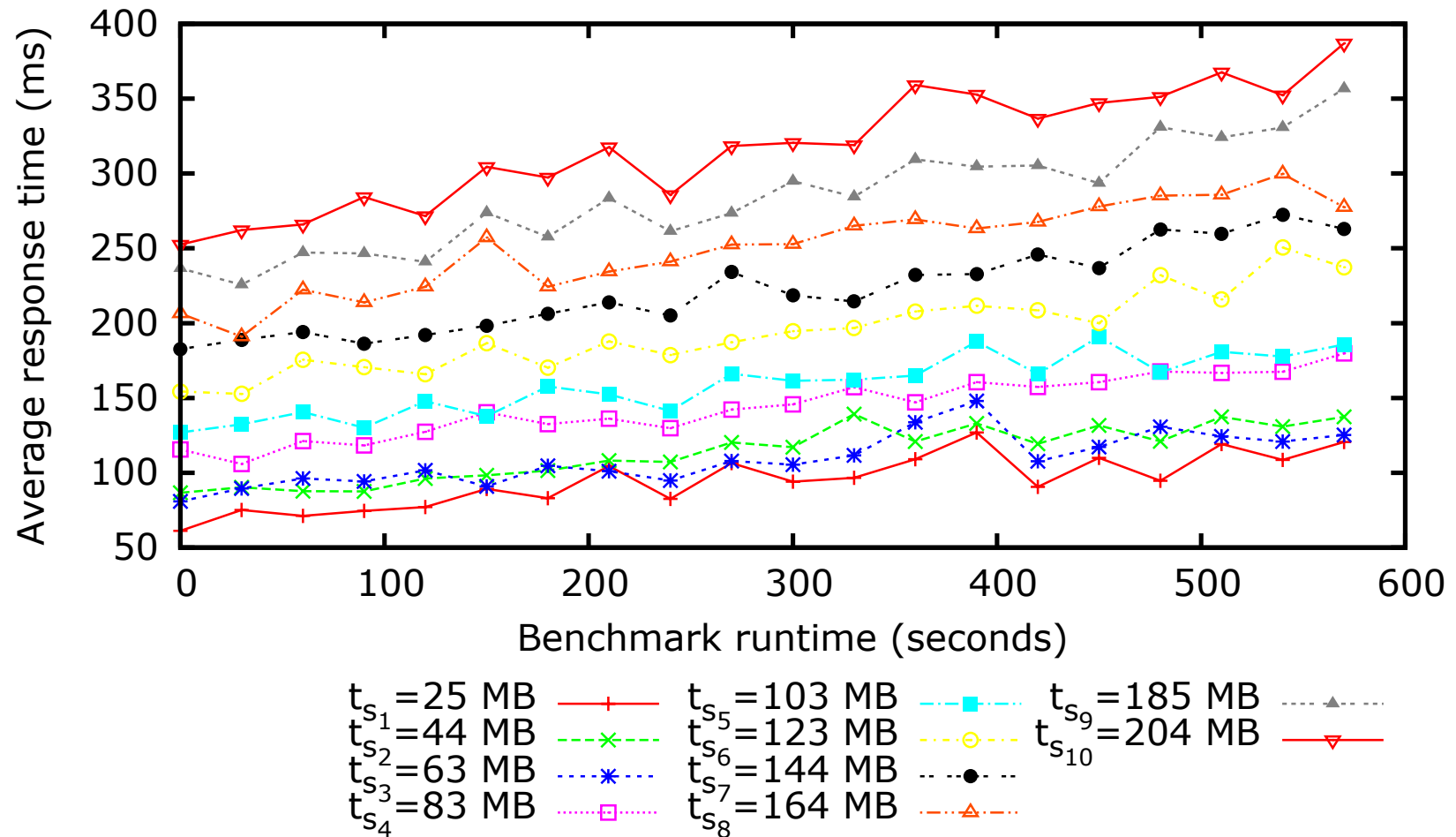
- Capacity degrades to 85%
- RMSE is 67.31 ms

- Destination node:

- Capacity degrades to 82%
- RMSE is 77.63 ms

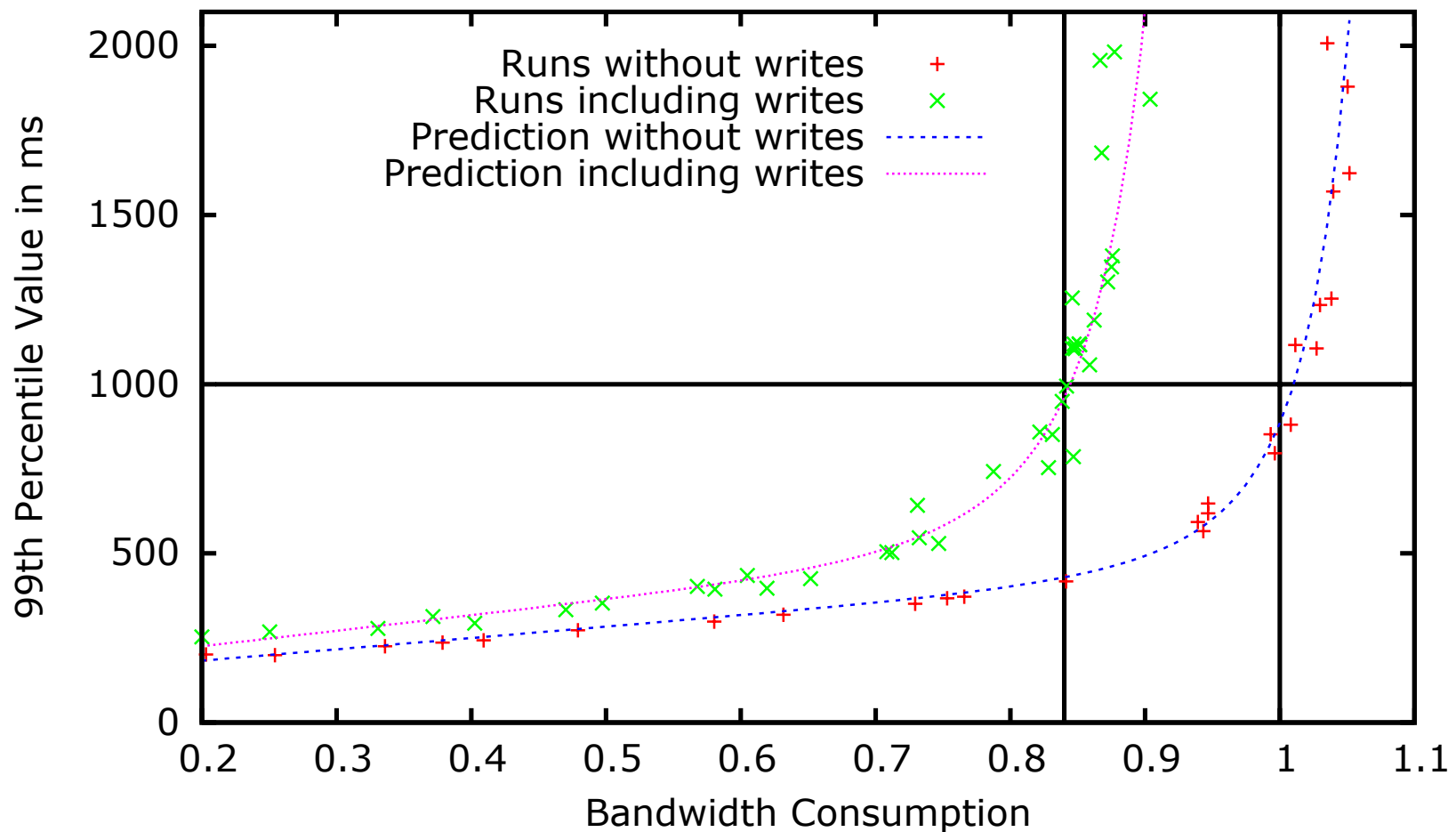
Extending the model with batch writes

- Writes are handled by delta buffer
- Increasing delta size leads to higher response times



Extending the model with writes

- Batch writes decrease a server's request processing capacity to 84%
- Root mean squared error for predicting values < 1000 ms is 62.36 ms



Automatic tenant migration algorithm

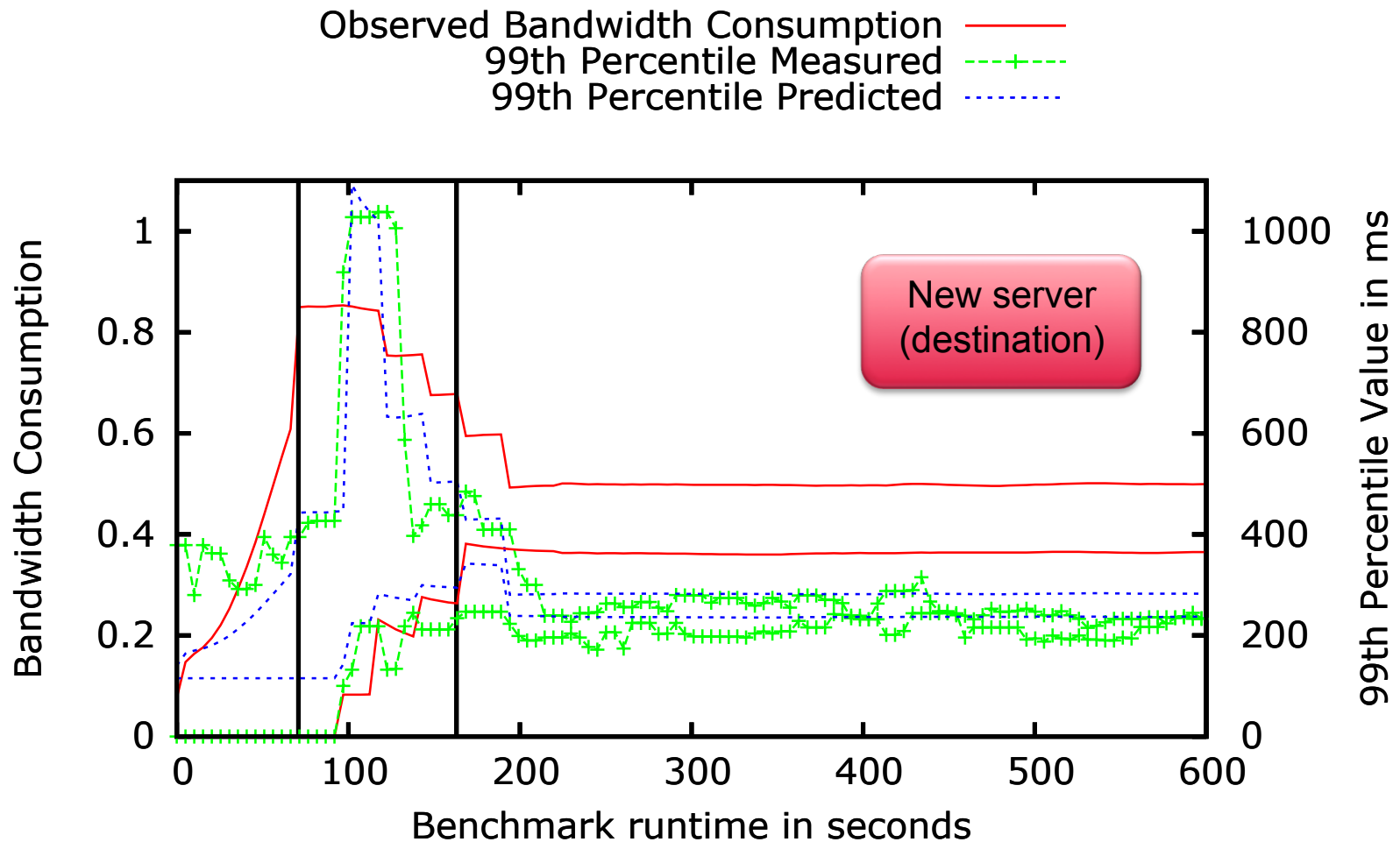
- Cluster leader continuously monitors workload on all servers and makes incremental changes to the tenant placement

- Handling overloaded server:
 - Server is considered overloaded at $b \geq 0.85$ (maximum for migrating tenants away from server without SLO violation)
 - Identify set of tenants to move away (select small tenants first)
 - Sort move set by bandwidth consumption (higher values first)

- Handling under-utilized server:
 - Server considered under-utilized at $b \leq 0.3$ (lower values result in too many migrations)
 - All tenants on server are in move set
 - If not possible to clear server then treat as a new server in overload case

Handling an overloaded server

- Workload increases up to $b=0.85$, algorithm is triggered
- Four tenants are migrated away and 99th percentile value decreases



Fourth question

Why should data in the cluster be replicated and how to do it?

Why is it good to have multiple copies of the data?

- Scalability beyond a certain number of concurrently active users
- High availability during normal operations
- Alternating execution of resource-intensive operations (e.g. merge)
- Rolling upgrades without downtime
- Data migration without downtime

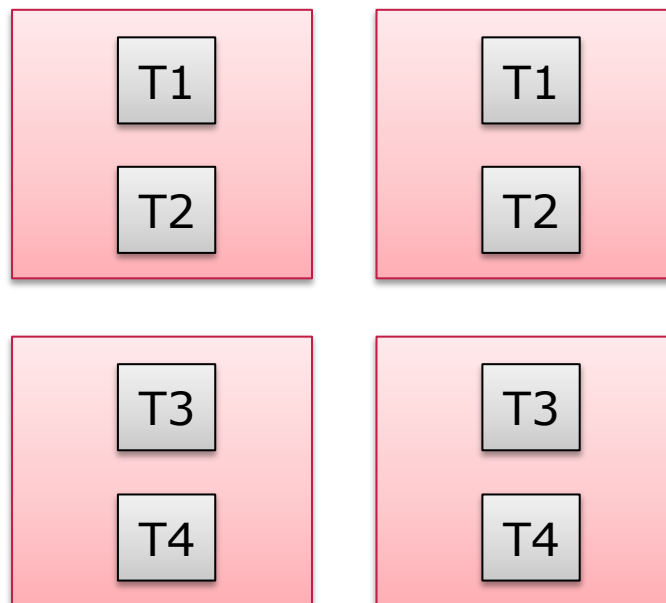
- *Reminder:* Two in-memory copies allow faster writes and are more predictable than one in-memory copy plus disk

- **Downsides:**
 - Response time goal might be violated during recovery
 - You need to plan for twice the capacity

Really?

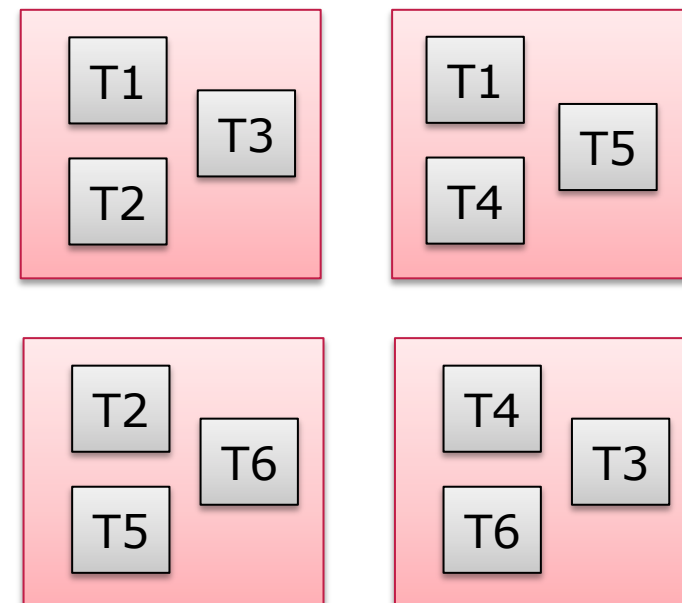
Tenant placement

Conventional Mirrored Layout



If a node fails, all work moves to one other node. The system must be **100% over-provisioned**.

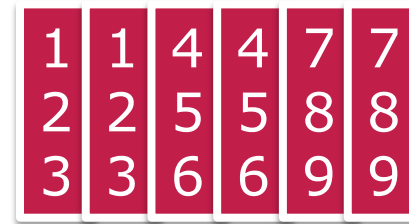
Interleaved Layout



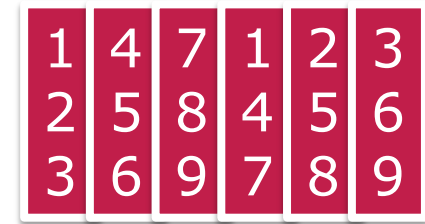
If a node fails, work moves to many other nodes. Allows **higher utilization** of nodes.

Interleaved tenant placement

- Perfect placement:
 - 100 tenants
 - 2 copies/tenant
 - All tenants have same size
 - 10 tenants/server



Mirrored



Interleaved

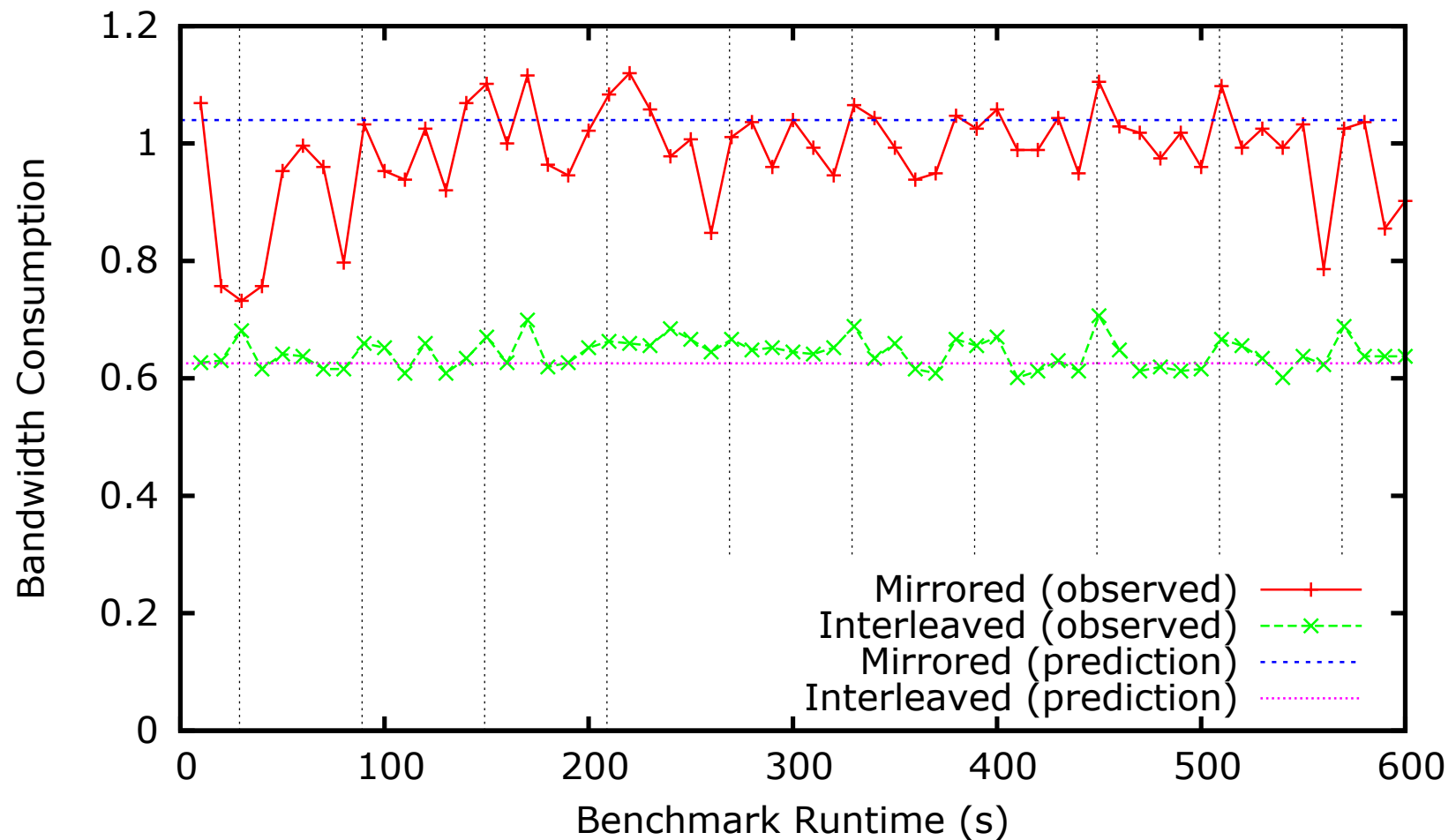
- Perfect balancing (same load on all tenants):
 - 6M rows (204 MB compressed) of data per tenant
 - The same (increasing) number of users per tenant
 - No writes

	Mirrored	Interleaved	Improvement
No failures	4218 users	4506 users	7%
Periodic single failures	2265 users	4250 users	88%

Throughput before violating response time goal

“Worst” server workload during failure

- Same tenant mix and request rates in mirrored vs. interleaved setup
- Multiple servers assume role of worst server due to round-robin failures



Requirements for placement algorithm

- An optimal placement algorithm needs to cope with multiple (conflicting) goals:
 - Balance load (b) across servers
 - Achieve good interleaving

- Use migrations consciously for online layout improvements (no big bang cluster re-organization)

- Take usage patterns into account
 - Request rates double during last week before end of quarter
 - Time-zones, Christmas, etc.

Conclusion

- Answers to four questions:

- ✓ How do “in-memory” and “cloud computing” fit together?
- ✓ Does virtualization have a negative impact on in-memory databases?
- ✓ How to predict response times of an in-memory column database?
- ✓ Why should data in the cluster be replicated and how to do it?

- **Questions?**

Thank you!

We have student jobs available:

jan.schaffner@hpi.uni-potsdam.de