Karlsruhe Institute of Technology

Towards Comprehensive Measurement of Consistency Guarantees for Cloud-Hosted Data Storage Services

David Bermbach (KIT)
Liang Zhao, Sherif Sakr (NICTA & UNSW)

AIFB, eOrganization Research Group

Motivation

- Developing applications on top of only eventually consistent cloud storage services is difficult and requires a change in mindsets
- Most applications can tolerate a certain degree of uncertainty as long as they know about the extent
- In some cases eventual consistency is just not acceptable but changing the behavior of storage systems requires knowledge of their guarantee interna
- Consistency guarantees are affected by additional influence factors (e.g., failures, workload, geo-distribution, …); this should be analyzed in a comprehensive consistency benchmark.

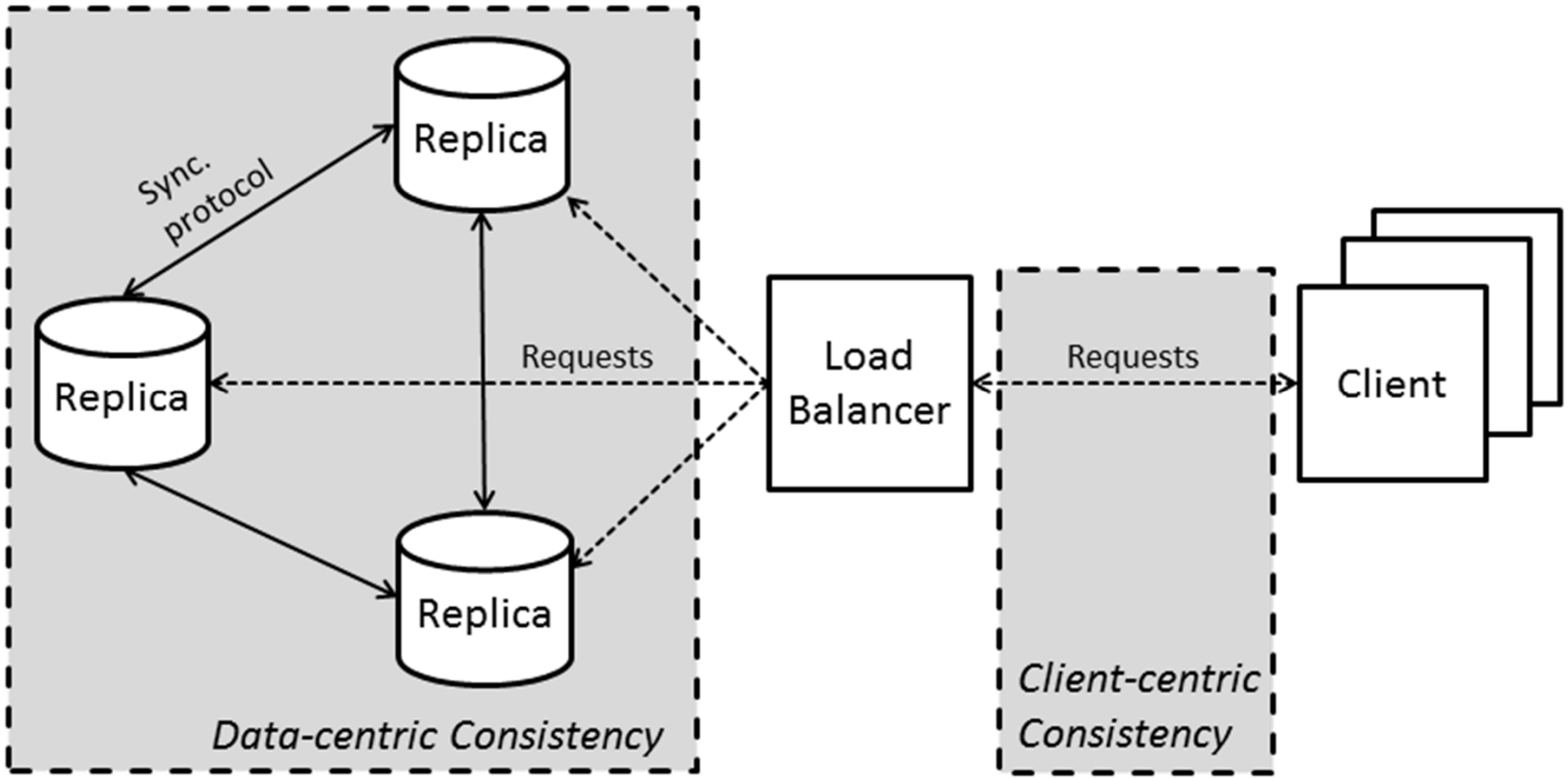
What is consistency?

- Two different definitions
- “C“ from ACID is not what distributed systems see as consistency

- Basic idea: If all replicas are identical and all ordering guarantees of the consistency model are observed, then we call the datastore *consistent*
- *Eventual Consistency*: In the absence of updates all replicas eventually converge towards a consistent state

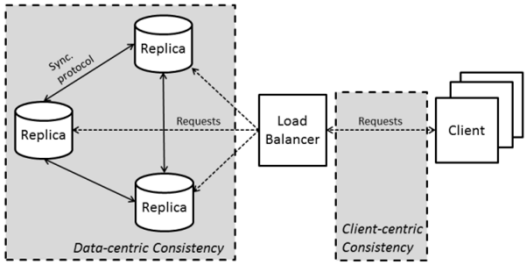
What is consistency? (cont.)

- Two perspectives: data-centric vs. client-centric



What is consistency? (cont.)

- Two perspectives: data-centric vs. client-centric



- Two dimensions:
  - Staleness: How far are replicas lagging behind
  - Ordering: How many requests are executed out of order

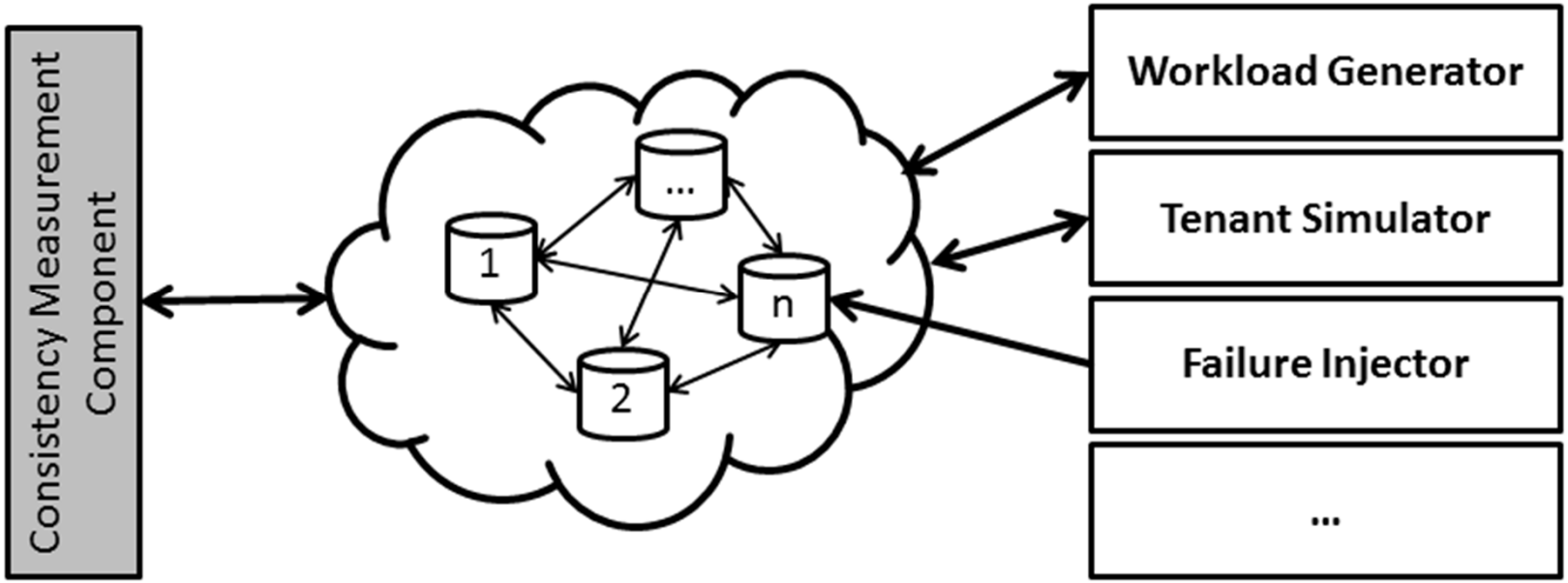
Metrics

- Staleness
  - In terms of time (t-Visibility)
  - In terms of versions (k-Staleness)
- Ordering
  - Probability of violations of Monotonic Read Consistency
  - Probability of violations of Monotonic Write Consistency
  - Probability of violations of Read Your Writes Consistency

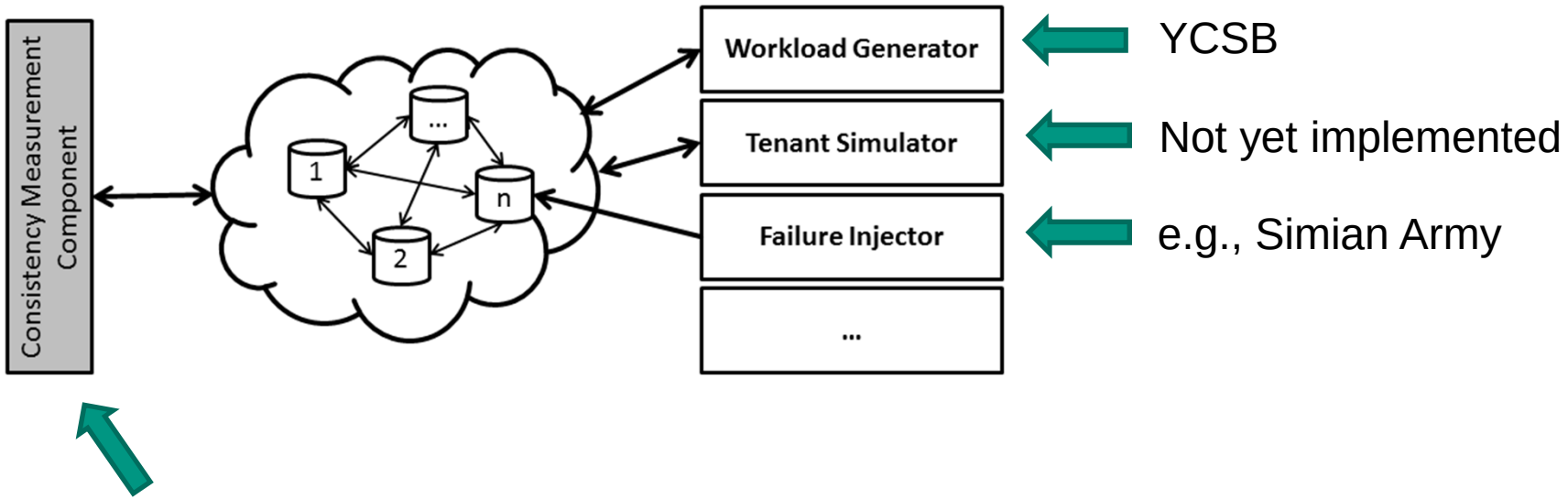
Challenges

- Accuracy and Meaningfulness
  - Choose appropriate metrics
  - Consider accuracy of the measurement approach
- Workloads
  - System load
  - Actual measurement workload (highly volatile reaction, so keep it simple)
- Geo-replication
  - Single site vs. multi site deployments vs. anything in between
- Failures
  - Cloud service vs. self-hosted system
  - Crash-stop, crash-recover, byzantine
- Multi-tenancy
  - Cross-effects between tenants

Architecture



Architecture



Based on Bermbach & Tai, MW4SOC 2011

Experiments

- Study effects of
  - different workloads
  - geo-distribution
- on consistency guarantees of
  - Cassandra: (3,1,1) quorum
  - MongoDB: Master-Slave

Setup

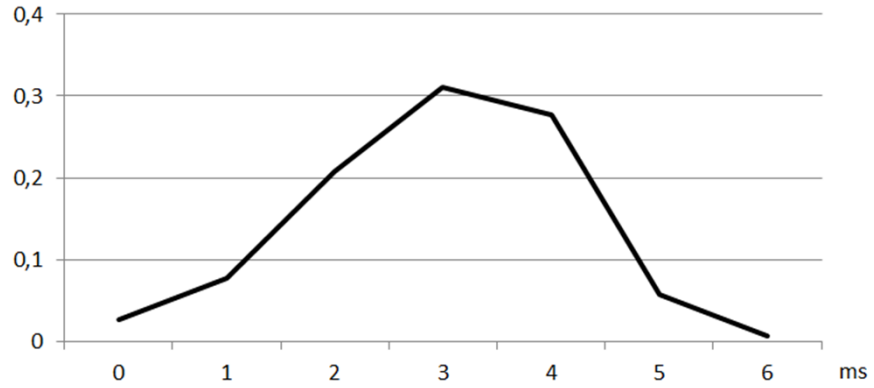
- Storage system deployed on 3 medium EC2 instances running in one AZ (eu-west-1a), multiple AZ (eu-west) or multiple regions (eu-west, us-west, asia-pacific)
- YCSB running on xlarge instance
- Consistency measurement running on 5 small instances per test
- Load balancer routes requests to the closest replica
- Measure consistency
  - Without additional workload
  - With 80% reads and 20% writes in parallel (YCSB)
  - With 20% reads and 80% writes in parallel (YCSB)

Staleness Results for Cassandra

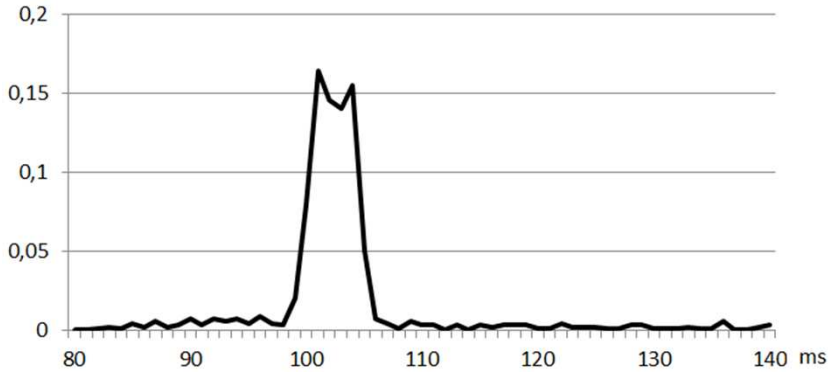
Single-AZ

<1ms for 98% of requests

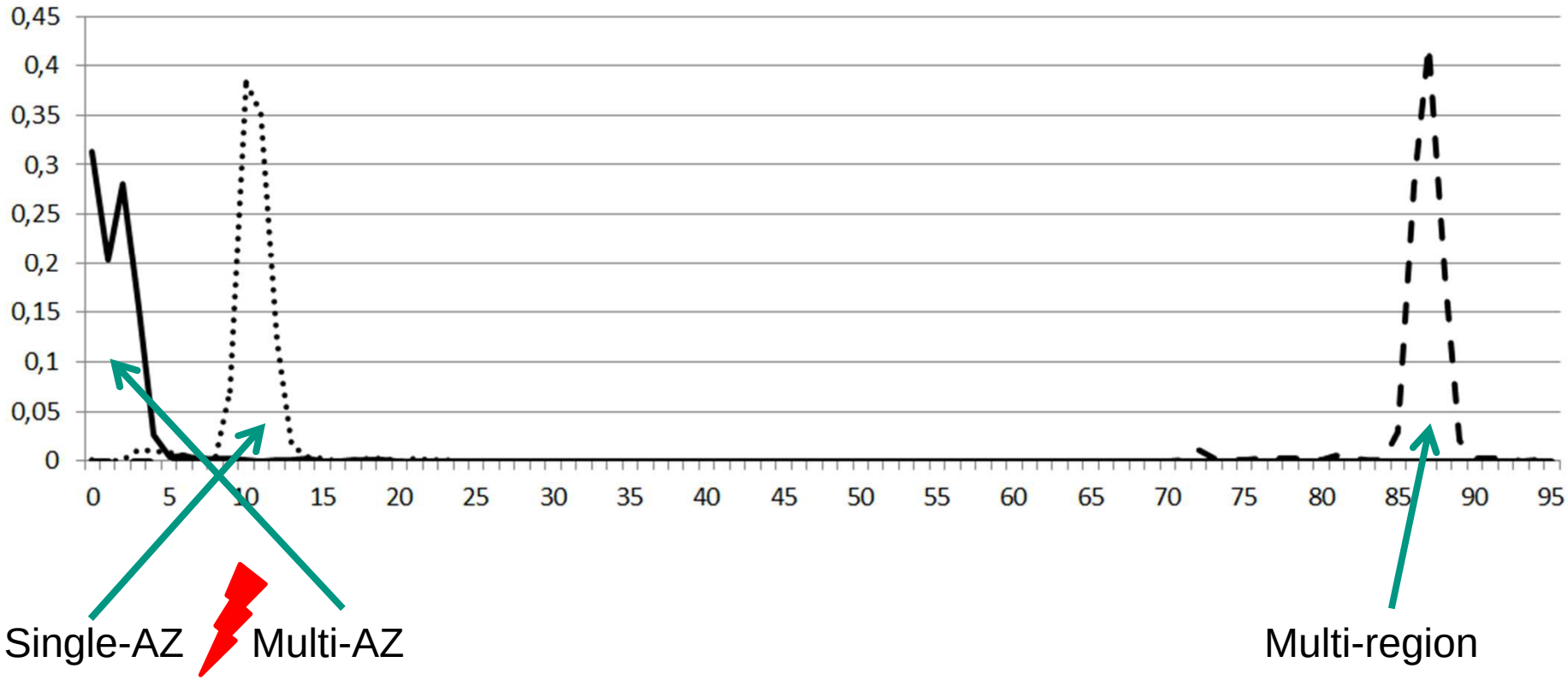
Multi-AZ



Multi-region

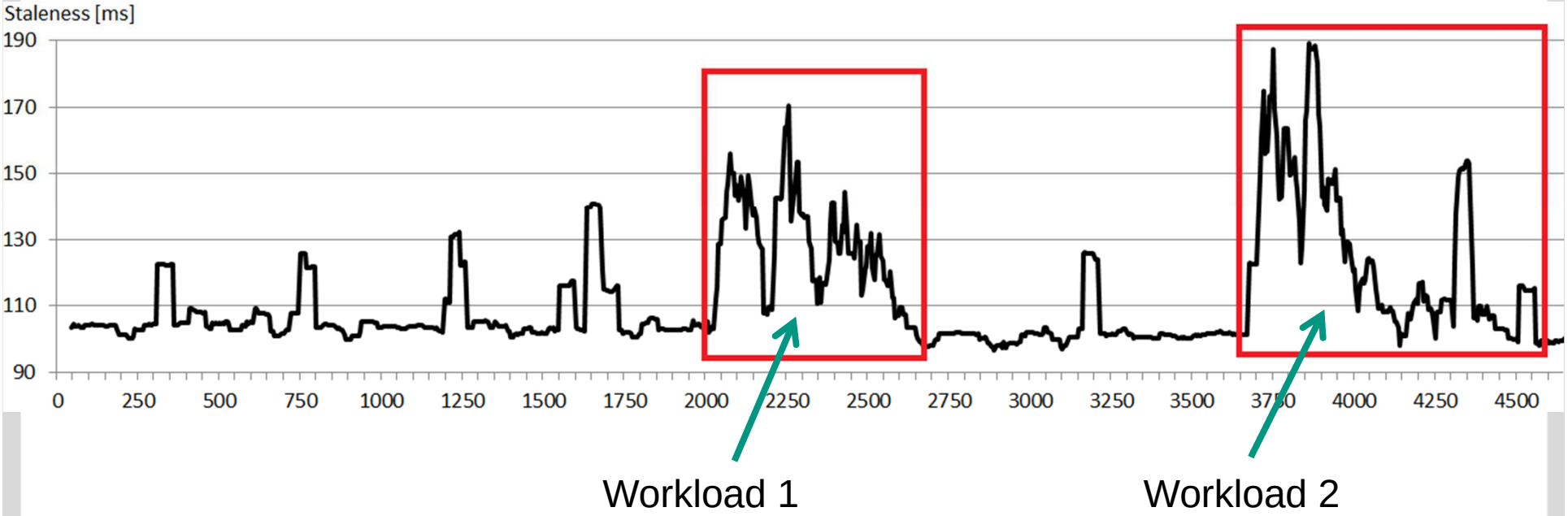


Staleness Results for MongoDB



Effects of parallel workload

- No effect except for multi-region Cassandra:



Conclusion

- A comprehensive consistency benchmark should
  - use meaningful and accurate (fine-grained & continuous) metrics
  - consider effects of workloads, geo-distribution, multi-tenancy and failures
  - preferably reuse and integrate popular, proven solutions
- Increased geo-distribution results in larger staleness values for both MongoDB and Cassandra
- Increased workloads seem to have little effect on consistency as long as CPU utilization is below 100%

Thank you for your attention

Questions?

david.bermbach@kit.edu

This paper was joint work with Liang Zhao and Sherif Sakr at NICTA and UNSW:
=> liang.zhao@nicta.com.au
=> sherif.sakr@nicta.com.au