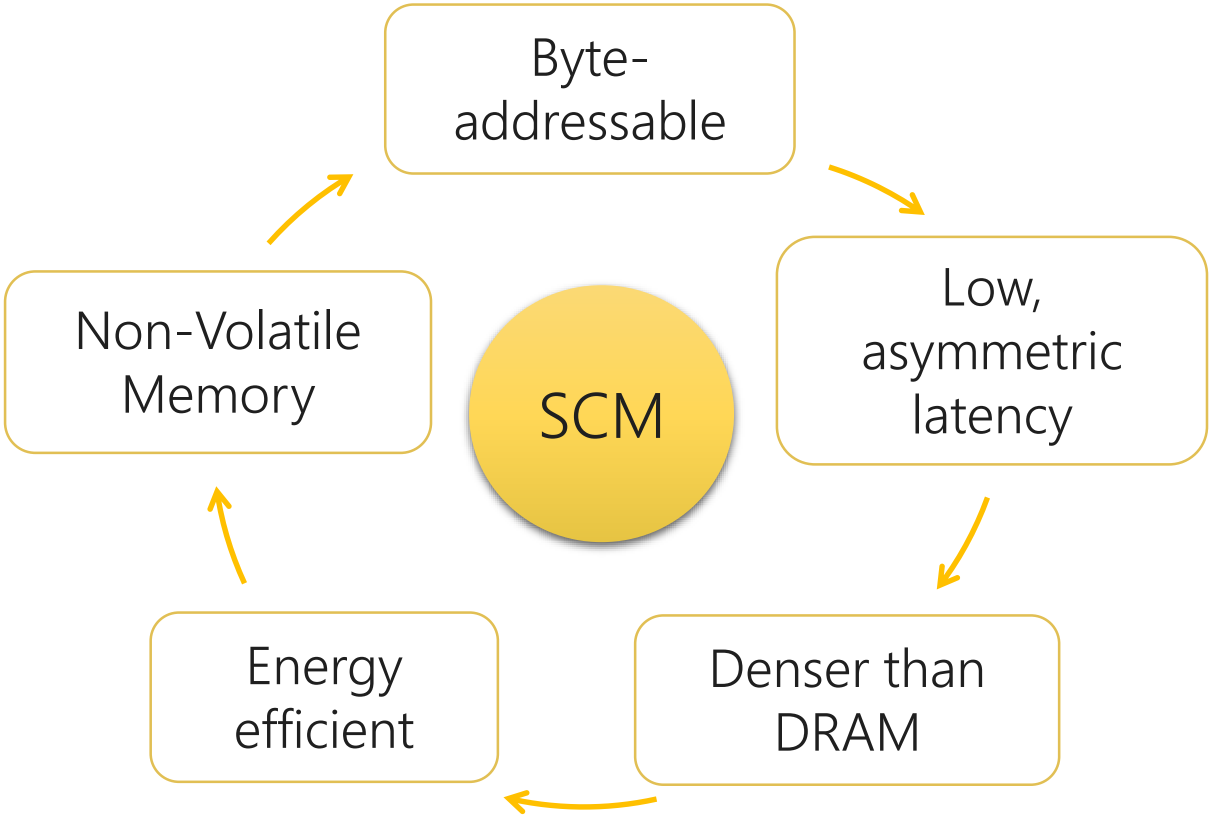
Instant Recovery for Main-Memory Databases

Ismail Oukid*°, Wolfgang Lehner*, Thomas Kissinger*, Peter Bumbulis°, and Thomas Willhalm +

*TU Dresden °SAP SE + Intel GmbH

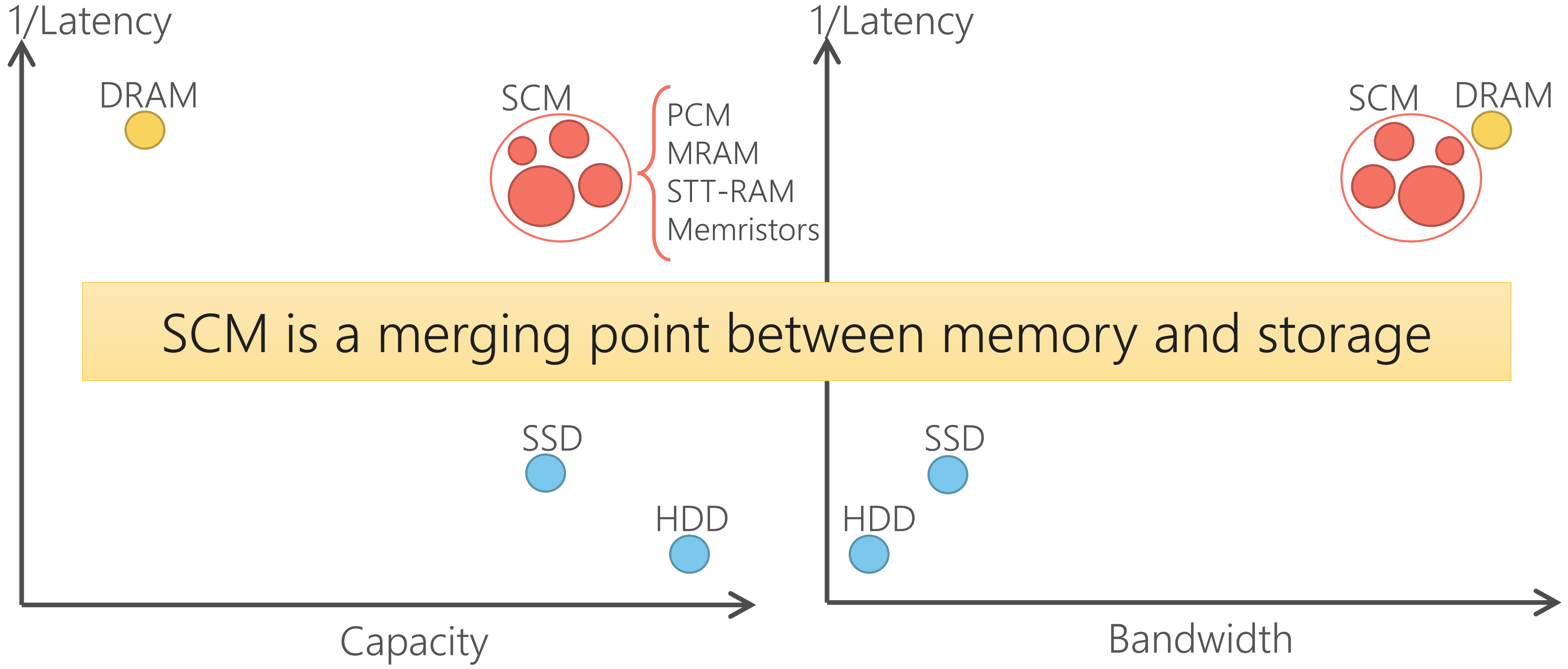
CIDR 2015, Asilomar, California, USA,
January 5, 2015

Storage Class Memory



SCM Compared with Today's Technologies

Dresden Database Systems Group



SCM is a merging point between memory and storage

SCM and Databases

Improving the logging infrastructure, e.g.:

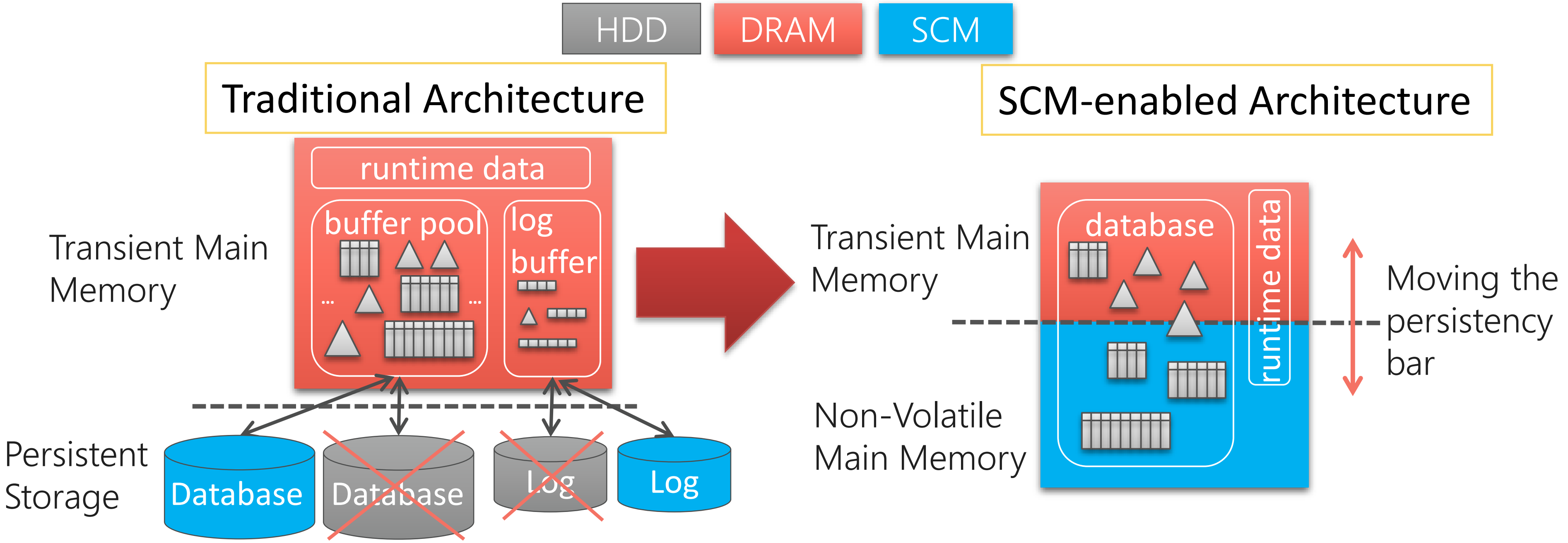
- Fang et al. High performance database logging using Storage Class Memory. ICDE'11
- Pelley et al. Storage management in the NVRAM era. VLDB'13
- Huang et al. NVRAM-aware Logging in Transaction Systems. VLDB'14

Improving specific database algorithms, e.g.:

- Chen et al. Rethinking Database Algorithms for Phase Change Memory. CIDR'11
- Stratis D. Viglas. Write-limited sorts and joins for persistent memory. VLDB'14

It takes a greenfield approach to measure the full potential of SCM

SCM-enabled Architecture



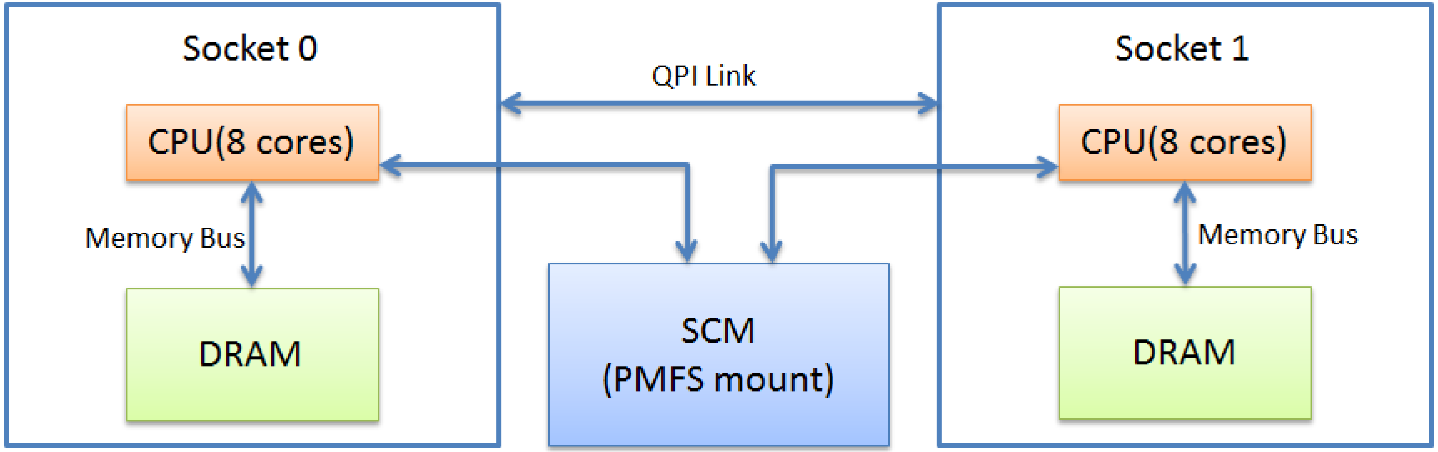
SOFORT is a single-level column-store, i.e., the working copy is the durable copy

Understanding SCM through Microbenchmarks

Dresden Database Systems Group

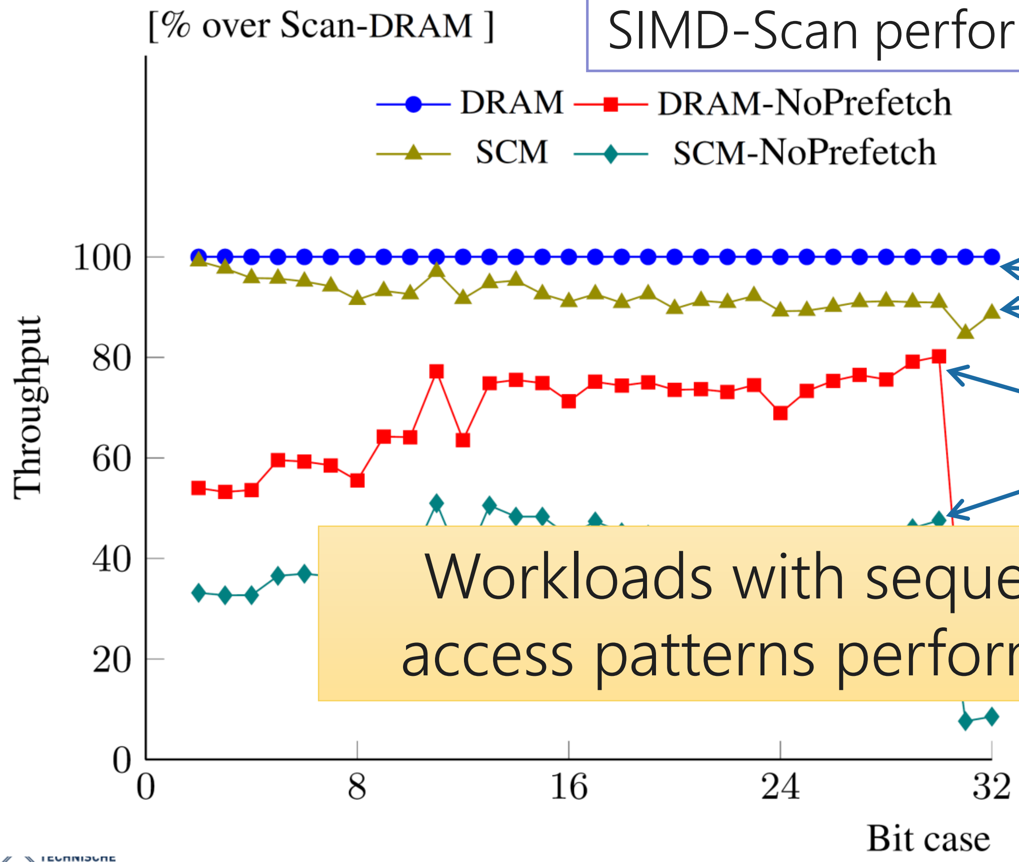
Hardware-based SCM simulation:

- Special BIOS, tunable latency with means of a microcode patch
- Limitation: symmetric instead of asymmetric read/write latency
- Avoiding NUMA effects: benchmark run on a single socket
- DRAM Latency: 90ns   SCM latency: 200ns



Understanding SCM through Microbenchmarks (2)

SIMD-Scan performance on DRAM and SCM



8% average slowdown

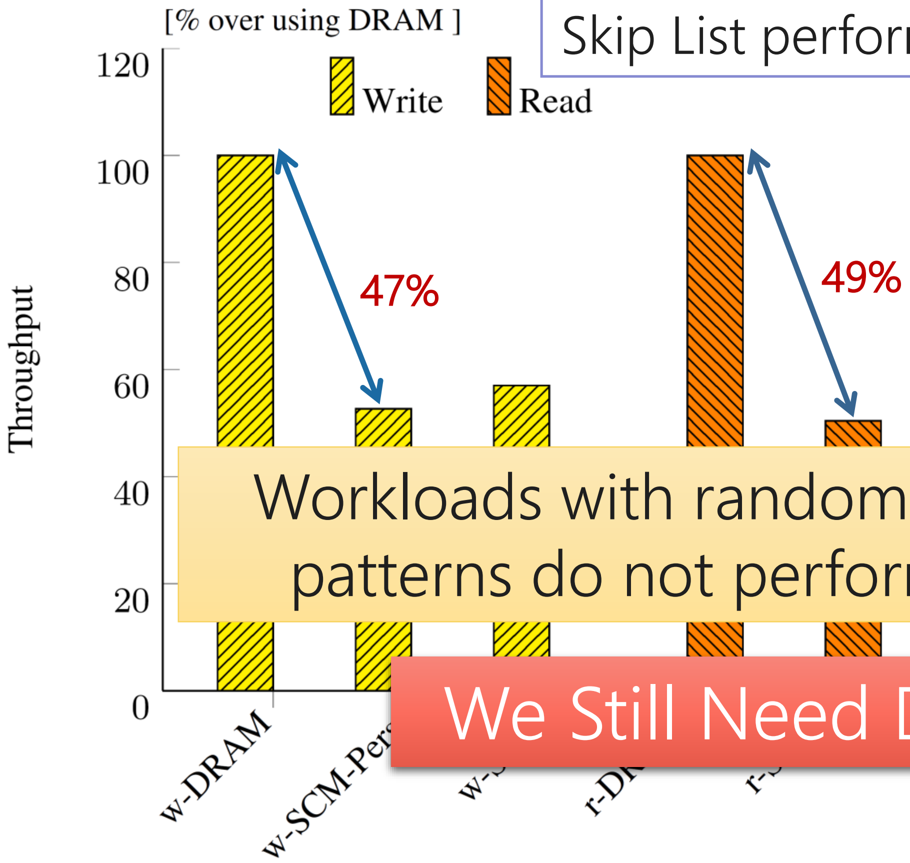
41% average slowdown

Workloads with sequential memory access patterns perform well on SCM

Understanding SCM through Microbenchmarks (3)

Dresden Database Systems Group

Skip List performance on DRAM and SCM

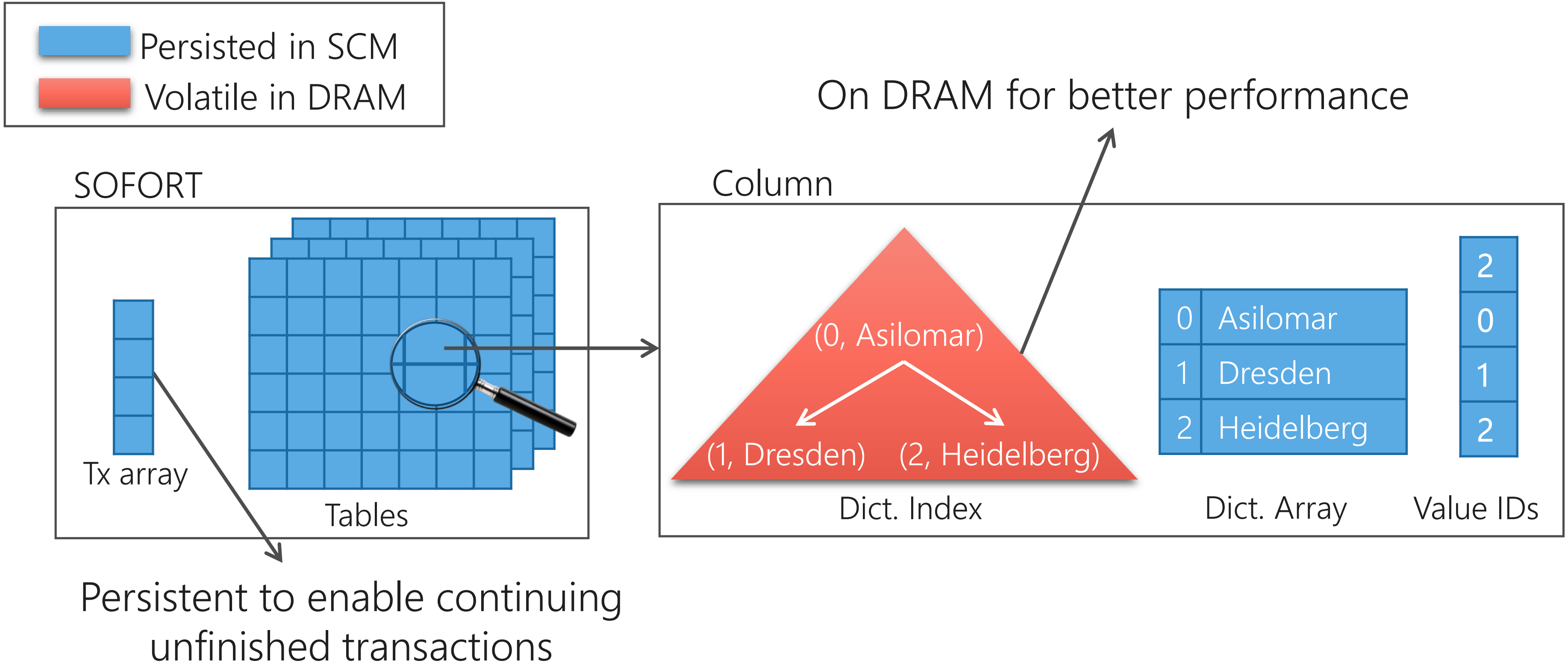


Workloads with random memory access patterns do not perform well on SCM

We Still Need DRAM

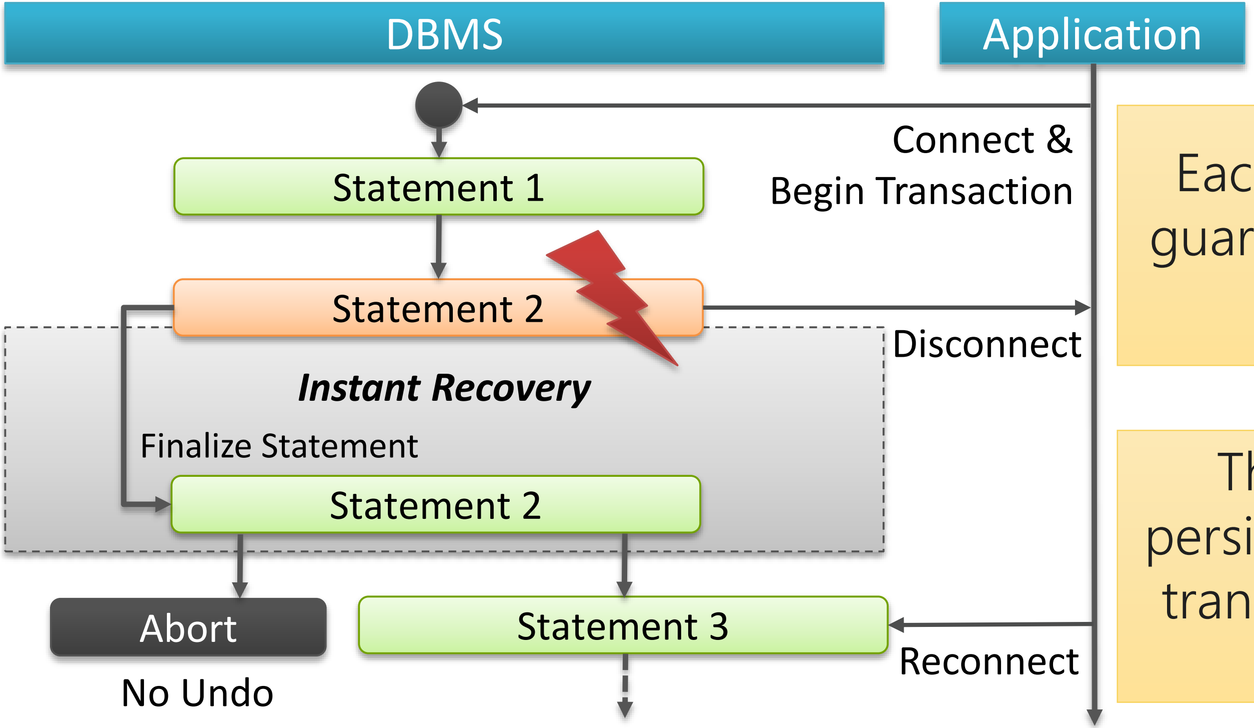
SOFORT Column Structure

Dresden Database Systems Group



Implementation details in “SOFORT: A Hybrid SCM-DRAM Storage Engine for Fast Data Recovery”, DaMoN’14

Continuing Unfinished Transactions

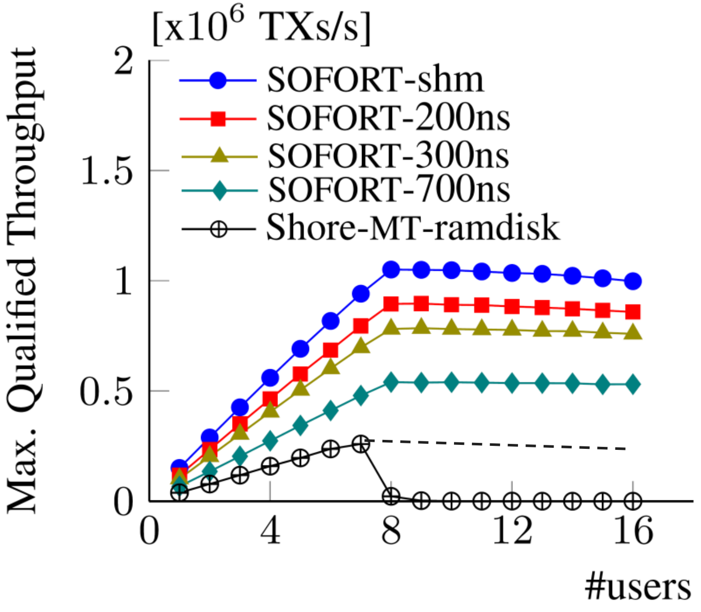


Each executed statement is guaranteed to have persisted its changes in SCM.

The Transaction array is persistent allowing unfinished transactions at crash time to continue.

Performance Overview

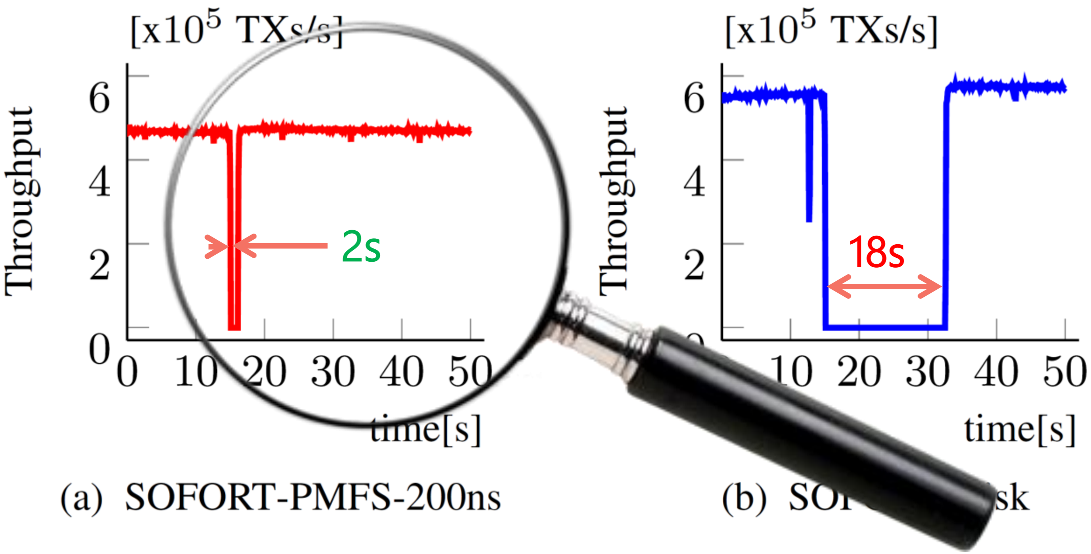
Throughput



TATP Mix

Competitive performance even in high latency environment

Restart Time



Fast restart time. No need to fetch data stored in SCM

Still not instant

Improving Recovery Performance

Dresden Database Systems Group

Synchronous Recovery

- Step 1: Recovery memory management
- Step 2: Recover primary data
- Step 3: Continue unfinished statements
- Step 4: Rebuild secondary data structures on DRAM
- Step 5: Start accepting user queries

Primary data already "loaded"

Restart time depends on the size of secondary data structures to be rebuilt

Instant Recovery

- Idea 1:
  - Use primary data to answer queries
  - Rebuild secondary data structures asynchronously

Instant responsiveness

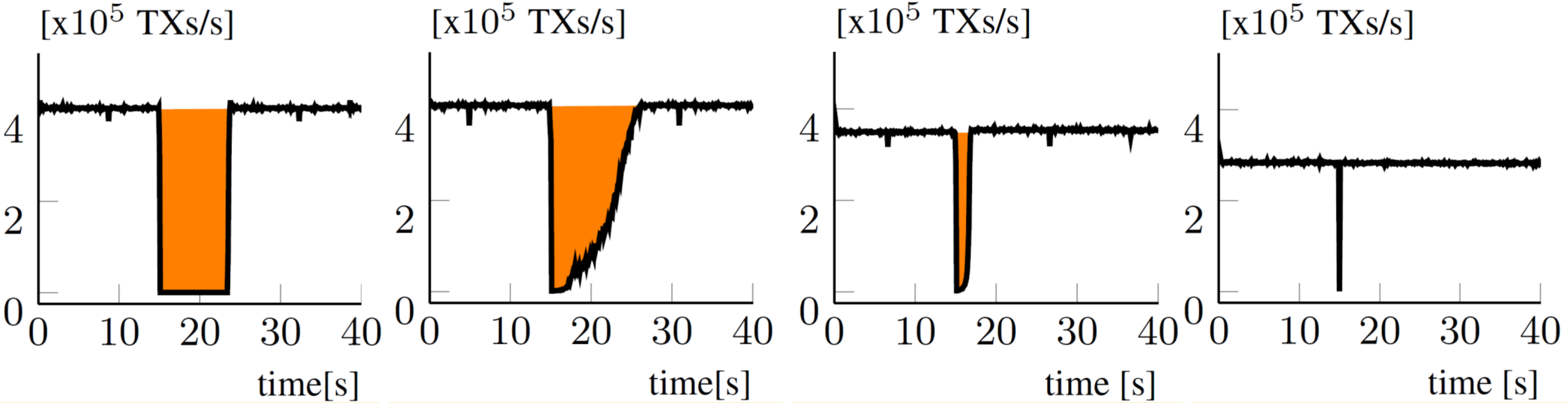
- Idea 2:
  - Persist part of or all secondary data structures in SCM

Instant recovery at peak performance

Perf. Penalty on throughput

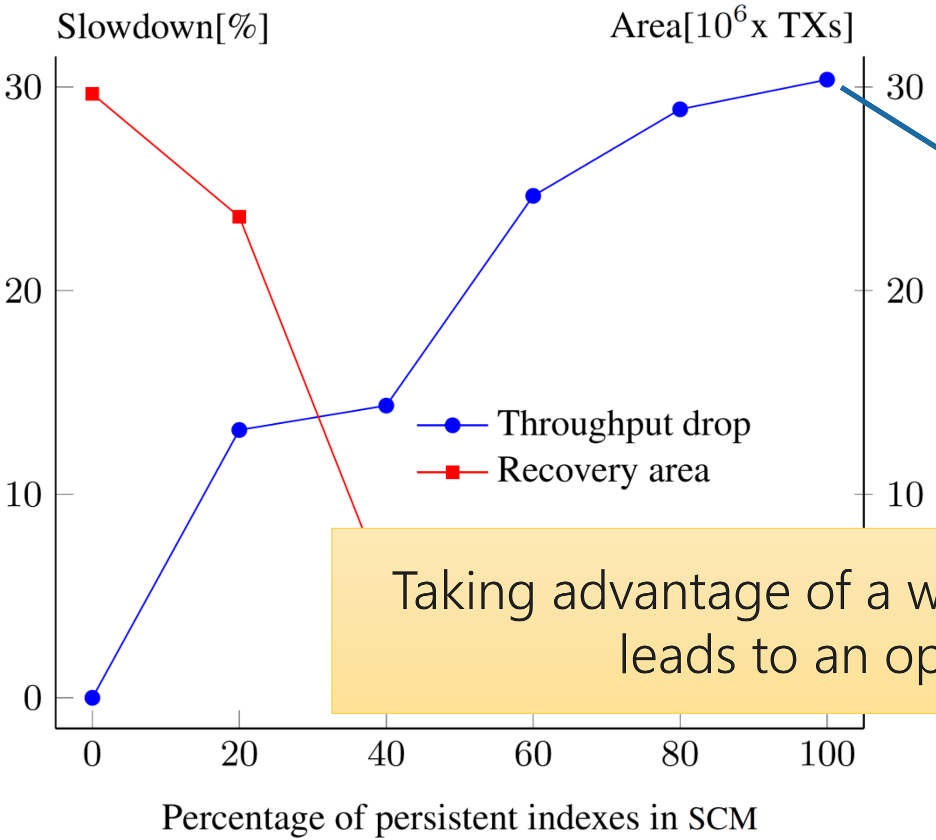
Evaluation: Recovery Time

Synchronous Recovery	Instant Recovery		
	0% indexes in SCM	40% indexes in SCM	100% indexes in SCM



Synchronous Recovery	0% indexes in SCM	40% indexes in SCM	100% indexes in SCM
First query accepted after ~8s, i.e., Recovery delta = 8s	Throughput: -0% Recovery area: -16% Recovery delta: ~8s	Throughput: -14% Recovery area: -82% Recovery delta: <2s	Throughput: -30% Recovery area: -99,8% Recovery delta: <5ms

Evaluation: Throughput Vs. Recovery

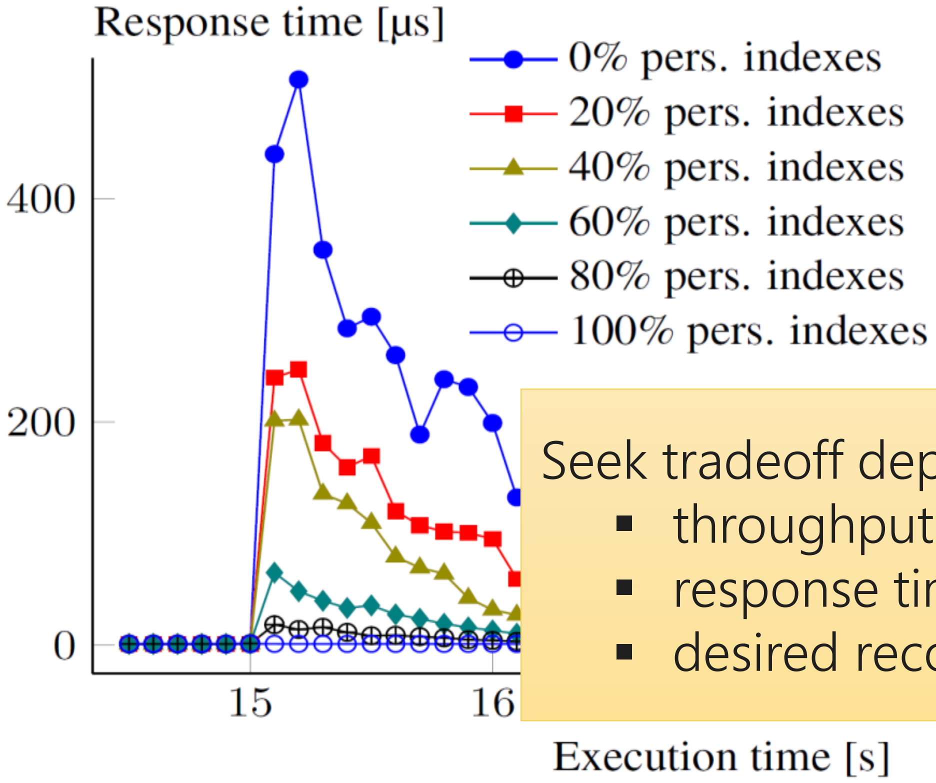


Throughput drop limited to 30%

Curves are not linear: secondary data structures are not equally important for TATP

Taking advantage of a workload's characteristics leads to an optimal tradeoff

Evaluation: Average Response Time



Max. avg. (over 100ms) Response time:
- 0% pers. indexes: **506µs**
- 100% pers. indexes: **2µs**

Seek tradeoff depending on:
- throughput requirements
- response time requirements
- desired recovery performance

Conclusion and Future Work

We showed that SCM can help:

- Achieve instant recovery for main-memory databases
- Continue unfinished transaction at crash time
- Simplify durability management
- Remove the need for a traditional transactional log

Current and future work include:

- Improve recovery performance without compromising query performance
- Design new SCM-friendly persistent indexing structures
- Persistent, DRAM like memory management for SCM
- Testing tools for single-level store architectures

Will SCM trigger a new rewrite of databases?

Thank You! Questions? Comments?

Ismail Oukid
ismail.oukid@sap.com
https://wwwdb.inf.tu-dresden.de/team/external-members/ismail-oukid/

Instant Recovery for Main-Memory Databases

Ismail Oukid*°, Wolfgang Lehner*, Thomas Kissinger*, Peter Bumbulis°, and Thomas Willhalm+

*TU Dresden °SAP SE + Intel GmbH