Who Watches the Watchmen?
On the Lack of Validation in NoSQL Benchmarking

Wolfram Wingerath, Steffen Friedrich, Felix Gessert, Norbert Ritter

University of Hamburg
Department of Informatics
Databases and Information Systems

March 5th, 2015
1. Benchmarking Tools need validation - but how ...?

2. Enter SickStore

3. Experimental Validation: YCSB++

4. NoSQLMark: Improving on YCSB++

5. References

6. Questions & Discussion
Motivation

Why Validate Benchmarking Tools?
Benchmarking NoSQL is Hard

Dimensions of Interest

- latency and throughput
- availability
  - direct
  - steady-state
- consistency
  - staleness (version-based, time-based)
  - ordering guarantees
  - durability
  - transactions
Existing Benchmarks

- Wada et al. [WFZ+11]
- Bermbach et al. [BT11, BT14, Ber14]
- YCSB++ [PPR+11]
- BG Benchmark [BG13]
- HP Labs [RGA+12, GLS11]
- YCSB+T [DFNR14]

General problems:

→ **no error bounds**

→ **often no validation**, i.e. you have to trust them...
Existing Benchmarks

An example: YCSB++

YCSB++

Writer

Reader

Zookeeper

1. subscribe for x

2. insert x

3. ack
Existing Benchmarks
An example: YCSB++

YCSB++
Writer

Zookeeper

YCSB++
Reader

1. subscribe for x

4. pub x

2. insert x

3. ack
Existing Benchmarks
An example: YCSB++
Existing Benchmarks
An example: YCSB++
Existing Benchmarks

An example: YCSB++
Existing Benchmarks

An example: YCSB++

possibly consistent between 7 and 10
but lower bound: \(6 - 5 = 1\)

Wingerath, Friedrich, Gessert, Ritter | Who Watches the Watchmen?
Existing Benchmarks

An example: YCSB++

possibly consistent between 7 and 10
but lower bound: 6 - 5 = 1
YCSB++ (no lower/upper bound): 11 - 5 = 6
Existing Benchmarks

An example: YCSB++

- YCSB++ Writer
- Zookeeper
- YCSB++ Reader

Network latency, even if writer and reader on same node.

1. subscribe for x
2. insert x
3. ack
4. pub x
5. go!
6. 7.
7. disrupted YCSB workload
8. no x
9. try to read x
10.

Possibly consistent between 7 and 10, but lower bound: 6 - 5 = 1
YCSB++ (no lower/upper bound): 11 - 5 = 6
Existing Benchmarks

But how bad is it?
Existing Benchmarks

But how bad is it?

hard to say, because measurements are taken on the client side only → actual system state is unknown
Single-Node Inconsistency to the rescue!

Enter SickStore
Single-Node Inconsistency to the rescue!

Multi-node

- Scalability: ✓
- Consistency: ✗

Single-node

- Scalability: ✗
- Consistency: ✓
Single-Node Inconsistency to the rescue!

Multi-node scalability

SickStore

Single-node consistency

scalability
Single-Node Inconsistency to the rescue!

- **SickStore**
  - Globally consistent logfile

- **Multi-node**
  - Scalability
  - Consistency

- **Single-node**
  - Globally consistent logfile

Wingerath, Friedrich, Gessert, Ritter | Who Watches the Watchmen?
Single-Node Inconsistency to the rescue!

Multi-node

Single-node

SickStore

scalability

consistency

Globally consistent logfile

Wingerath, Friedrich, Gessert, Ritter | Who Watches the Watchmen?
SickStore
Single-node inconsistent key-value Store

- **consistent Backend**: write requests are executed in their actual order (last-write-wins)
- **Multi-node Behaviour**: queries are served by virtual storage nodes subject to controlled anomalies
- **Tunable staleness**: the delay by which data become visible to storage nodes is configurable
- **Globally consistent logfile**: no clock drift between storage nodes → unambiguous system state
1. write
   x:{val: Bob}

6. read
   x
   1return Alice

7. x:{tr:5, srv:B}

B.x:{tr:7, srv:C}

11. return
   Alice

12. read
   x
   13. x:{tr:7, srv:C}

17. return
    Bob

SickStore

Wingerath, Friedrich, Gessert, Ritter | Who Watches the Watchmen?
Experimental Validation

YCSB++

Wingerath, Friedrich, Gessert, Ritter | Who Watches the Watchmen?
Experimental Validation: YCSB++

staleness 1 sec fixed @ 5 ops/sec over 40 seconds
Experimental Validation: YCSB++

Staleness 1 sec fixed @ 5 ops/sec over 40 seconds
Experimental Validation: YCSB++

Staleness 1 sec fixed @ 5 ops/sec over 40 seconds

YCSB++ data-centric staleness

Actually exposed

Staleness (ms)
Experimental Validation: YCSB++
Not even by a long shot...
Experimental Validation: YCSB++

Not even by a long shot...
Experimental Validation: YCSB++

What went wrong?

Implementation error:
ZooKeeper queue basically was a stack!
Experimental Validation: YCSB++

What went wrong?

Implementation error:
Zookeeper queue basically was a stack!
Experimental Validation: YCSB++

What went wrong?

Implementation error:
Zookeeper queue basically was a stack!
Experimental Validation: YCSB++

What went wrong?

Implementation error:
Zookeeper queue basically was a stack!
Experimental Validation: YCSB++
What went wrong?

Implementation error:
Zookeeper queue basically was a stack!
Experimental Validation: YCSB++
What went wrong?

Implementation error:
Zookeeper queue basically was a stack!
Experimental Validation: YCSB++

What went wrong?

Implementation error:
Zookeeper queue basically was a stack!
Experimental Validation: YCSB++
What went wrong?

Implementation error: Zookeeper queue basically was a stack!
→ max. staleness = experiment duration!
→ arbitrary measurement error
→ staleness is detected, but not reasonably quantified
Next on our Agenda:

Valid Benchmarking with NoSQLMark
NoSQLMark: Improving on YCSB++

YCSB++ Writer

Possibly consistent between 7 and 10
but lower bound: \(6 - 5 = 1\)

YCSB++ (no lower/upper bound): \(11 - 5 = 6\)

Wingerath, Friedrich, Gessert, Ritter | Who Watches the Watchmen?
NoSQLMark: Improving on YCSB++

Wingerath, Friedrich, Gessert, Ritter | Who Watches the Watchmen?
NoSQLMark: Improving on YCSB++
NoSQLMark: Improving on YCSB++

Wingerath, Friedrich, Gessert, Ritter | Who Watches the Watchmen?
NoSQLMark: Improving on YCSB++

1. subscribe for x
lower bound: $6 - 5 = 1$
upper bound: $13 - 2 = 11$
**NoSQLMark**

**In a Nutshell**

- **embedded pub-sub**: reduced network communication
- **configurable disruption**: proportion, frequency etc. of consistency reads are tunable
- **unified workload**: no separation between workload and consistency measurement operations
- **lower and upper bounds** for staleness
- **experimental verification** through SickStore
References


Bermbach, David; Tai, Stefan: Benchmarking Eventual Consistency: Lessons Learned from Long-Term Experimental Studies. In: Proceedings of the 2nd IEEE International Conference on Cloud Engineering (IC2E), IEEE, 2014. – Best Paper Runner Up Award
References

Dey, Akon; Fekete, Alan; Nambiar, Raghunath; Röhm, Uwe: YCSB+T: Benchmarking Web-scale Transactional Databases. In: Proceedings of International Workshop on Cloud Data Management (CloudDB’14). Chicago, USA, 2014


Wada, Hiroshi; Fekete, Alan; Zhao, Liang; Lee, Kevin; Liu, Anna: Data Consistency Properties and the Trade-offs in Commercial Cloud Storage: the Consumers’ Perspective. In: CIDR’11, 2011, S. 134–143
Questions?