ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

TPC-BiH:
A Benchmark for Bitemporal Databases

Martin Kaufmann, Peter M. Fischer, Norman May,
Donald Kossmann

SAP

ETH Zürich

UNI FREIBURG

Motivation: Inventory Management

OrderedItems

OrderID	ProdID	Ship	Time
1	mbpro	false	10 – ∞
1	w530	false	16 – 19
2	mbair	false	17 – ∞
1	w530	true	19 – ∞

⋈

Inventory

ProdID	Price	Time
w530	3000	1 – 15
mbpro	2000	1 – 15
mbair	2500	1 – ∞
w530	3050	15 – ∞
mbpro	2100	15 – 17
mbpro	1900	17 – ∞
x220	2300	20 – ∞

Analysing temporal data

1) What was the total value of the inventory at each point in time?
   - *Temporal Aggregation*
2) What was the price of MacBook Pro on January 15th?
   - *Time-Travel*
3) What was the most expensive order last year?
   - *Temporal Join*

Motivation: Why do we need another benchmark?

- Increasing demand for temporal features
  - Dozens of apps require it
  - Adoption of bitemporal features in SQL:2011
  - Support by commercial database vendors

- Users need to assess performance
  - Keeping and querying history not reflected by existing benchmarks
  - Temporal processing is performance-critical
  - Understanding of the performance characteristics of alternative implementations of temporal operators

Outline

- Motivation
- **State-of-the-art**
- The TPC-BiH Benchmark
- Preliminary Results
- Conclusion

Temporal Data: State of the Art

- Long-Running Research
  - Access Methods for Multiversion Data. David Lomet et al. 1989
  - TSQL Language Specification. Richard Snodgrass et al. 1994

- Currently limited support on language side
  - SQL:2011 is a big step forwards
  - No complete coverage of all use cases

- Commercial systems start adding temporal features
  - Bitemporal data: Oracle, Teradata, DB2 (SQL:2011)
  - System time: SAP HANA

Existing Temporal Benchmarks

- Previous work on benchmarking the temporal dimension:
  - Requirements specifications (Dunham et al., 1995)
  - Functional tests (test suite from the TSQL2 editors)

- Recent activity and closest match:
  "Adding a Temporal Dimension to the TPC-H Benchmark" (TPCTC 2012)
  - Limited extension of TPC-H schema
  - Sketches of data and queries
  - Focus on TSQL2

Outline

- Motivation
- State-of-the-art
- **The TPC-BiH Benchmark**
- Preliminary Results
- Conclusion

Methodology of the Benchmark

Overall goals

- Comprehensive benchmark for bitemporal query processing
- Benchmark settings reflect
  - real-life customer workloads
  - synthetic tests of temporal properties
- Benchmark is targeted towards SQL:2011

Schema

- Extend TPC-H tables with different types of history classes:
  - fully bitemporal
  - degenerated
  - multiple user times

Methodology of the Benchmark (II)

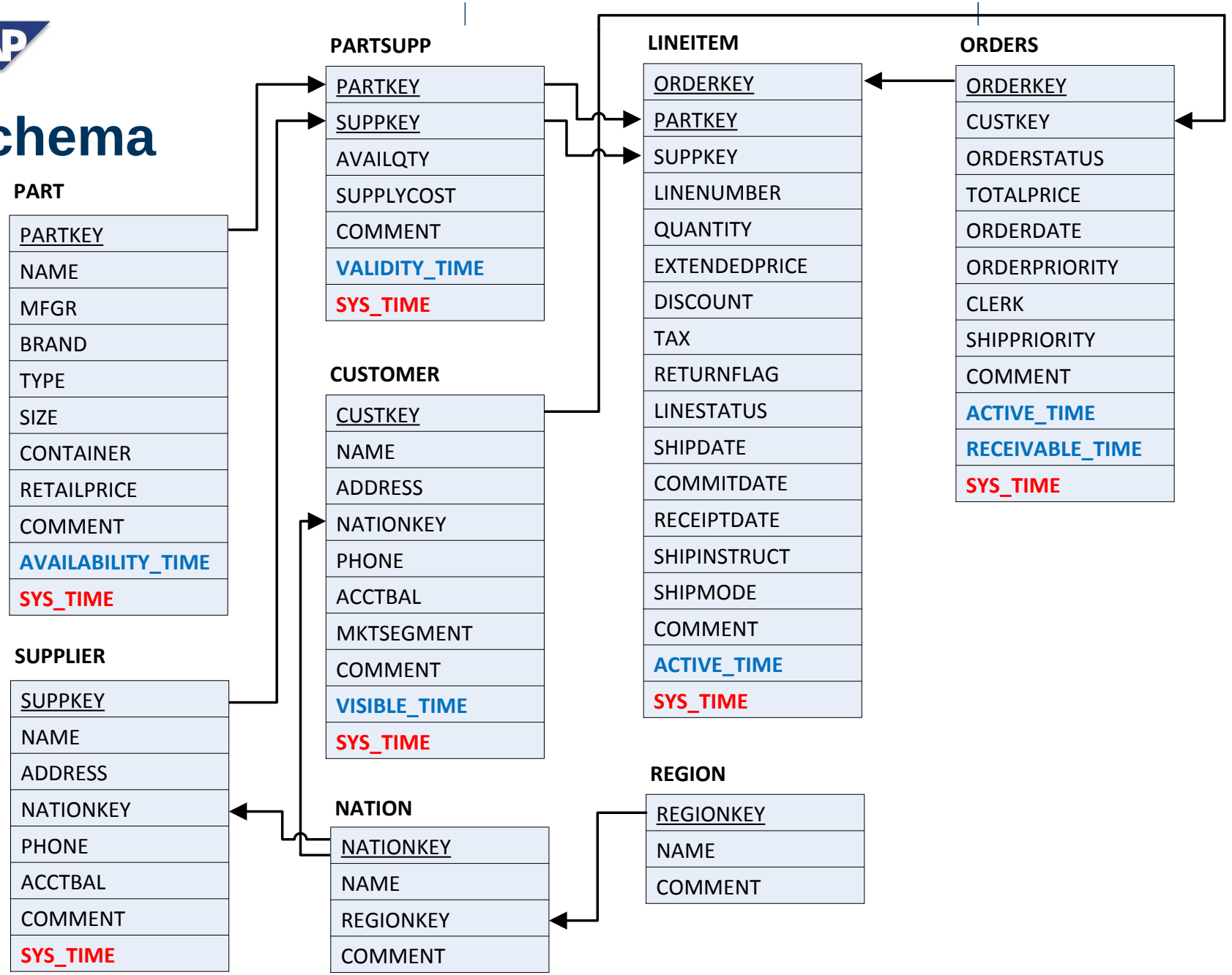
Data

- Initialization
  - Initial version is regular TPC-H data
  - Derive application time from TPC-H data semantics
- History
  - Evolve through carefully designed update scenarios
  - Maintain overall distributions at each point in tinme

Queries

- Coverage of common temporal DB requirements
- Stressing the system for individual time dimensions
- Correlations among the dimensions whenever relevant

Schema


PART
PARTKEY
NAME
MFGR
BRAND
TYPE
SIZE
CONTAINER
RETAILPRICE
COMMENT
AVAILABILITY_TIME
SYS_TIME
PARTSUPP
PARTKEY
SUPPKEY
AVAILQTY
SUPPLYCOST
COMMENT
VALIDITY_TIME
SYS_TIME
LINEITEM
ORDERKEY
PARTKEY
SUPPKEY
LINENUMBER
QUANTITY
EXTENDEDPRICE
DISCOUNT
TAX
RETURNFLAG
LINESTATUS
SHIPDATE
COMMITDATE
RECEIPTDATE
SHIPINSTRUCT
SHIPMODE
COMMENT
ACTIVE_TIME
SYS_TIME
ORDERS
ORDERKEY
CUSTKEY
ORDERSTATUS
TOTALPRICE
ORDERDATE
ORDERPRIORITY
CLERK
SHIPPRIORITY
COMMENT
ACTIVE_TIME
RECEIVABLE_TIME
SYS_TIME
CUSTOMER
CUSTKEY
NAME
ADDRESS
NATIONKEY
PHONE
ACCTBAL
MKTSEGMENT
COMMENT
VISIBLE_TIME
SYS_TIME
SUPPLIER
SUPPKEY
NAME
ADDRESS
NATIONKEY
PHONE
ACCTBAL
COMMENT
SYS_TIME
NATION
NATIONKEY
NAME
REGIONKEY
COMMENT
REGION
REGIONKEY
NAME
COMMENT

Data Generator

- Use TPC-H `dbgen` for version 0
- Generate history by adding 9 realistic update scenarios:

Scenario	Updated Tables	Probability
New Order	Customer, Orders, Lineitem	0.3
Cancel Order	Orders, Lineitem	0.05
Deliver Order	Customer, Orders, Lineitem, Partsupp	0.25
Receive Payment	Orders, Customer	0.20
Update Stock	Partsupp	0.05
Delay Availability	Part	0.05
Change Supplier Price	Partsupp	0.05
Update Supplier	Supplier	0.049
Manipulate Order	Orders, Lineitems	0.01

Queries

- The queries cover operations such as time travel, key in time, temporal joins and temporal aggregation

- Investigate many patterns of storage access and time- vs. key-oriented access with varying ranges and selectivity

- **Classes of TPC-BiH benchmark queries:**
  1) Pure-Timeslice Queries (Time Travel)
  2) Pure-Key Queries (Audit)
  3) Range-Timeslice Queries
  4) Bitemporal Queries

1) Pure-Timeslice Queries (Time Travel)

Motivation

- Testing "slices" of time, i.e., state of table at a previous time
- Each time dimension can be treated as a point or slice
  - Point: looking at a single time
  - Complete slice: full evolution of a tuple through a time dimension
- Evaluate on application time, system time or both

Example: "Value of all unshipped orders at a given time"

```
SELECT SUM(o_totalprice) AS revenue
FROM ORDERS
FOR SYSTEM_TIME AS OF TIMESTAMP '[TIME1]'
FOR BUSINESS_TIME AS OF '[TIME2]'
WHERE o_orderstatus = 'O'
```

2) Pure-Key Queries (Audit)

Motivation

- History of a single or small set of tuples
- Investigate how tuples evolve over time
- Evolution along the system time, application time(s) or both

Example: "What is the history of a given order?"

```
SELECT orderstatus, totalprice, orderpriority,
    shippriority, sys_time_start
FROM orders
FOR SYSTEM_TIME FROM '0001-01-01' TO '9999-12-30'
WHERE o_orderkey = [ORDER_KEY]
ORDER BY sys_time_start
```

3) Range-Timeslice Queries

Motivation

- Permit any combination of constraints
- General access pattern: both value and temporal aspects

Example: "When did we have the largest number of unshipped items?"

```
CREATE VIEW unshipped AS
  SELECT COUNT(*) AS total, li.VERSION() as version
  FROM LINEITEM li
  WHERE li.L_LINESTATUS = 'O'
  GROUP BY li.VERSION();

SELECT version, total FROM unshipped
WHERE total = (SELECT MAX(total) FROM unshipped)
```

4) Bitemporal Queries

Motivation

- Stress several time dimensions at the same time
- Vary the usage of each time dimension
  - a) point in time: current vs. past
  - b) sequenced/time range
  - c) non-sequenced/agnostic of time
- Complementary query variants to cover all combinations

Outline

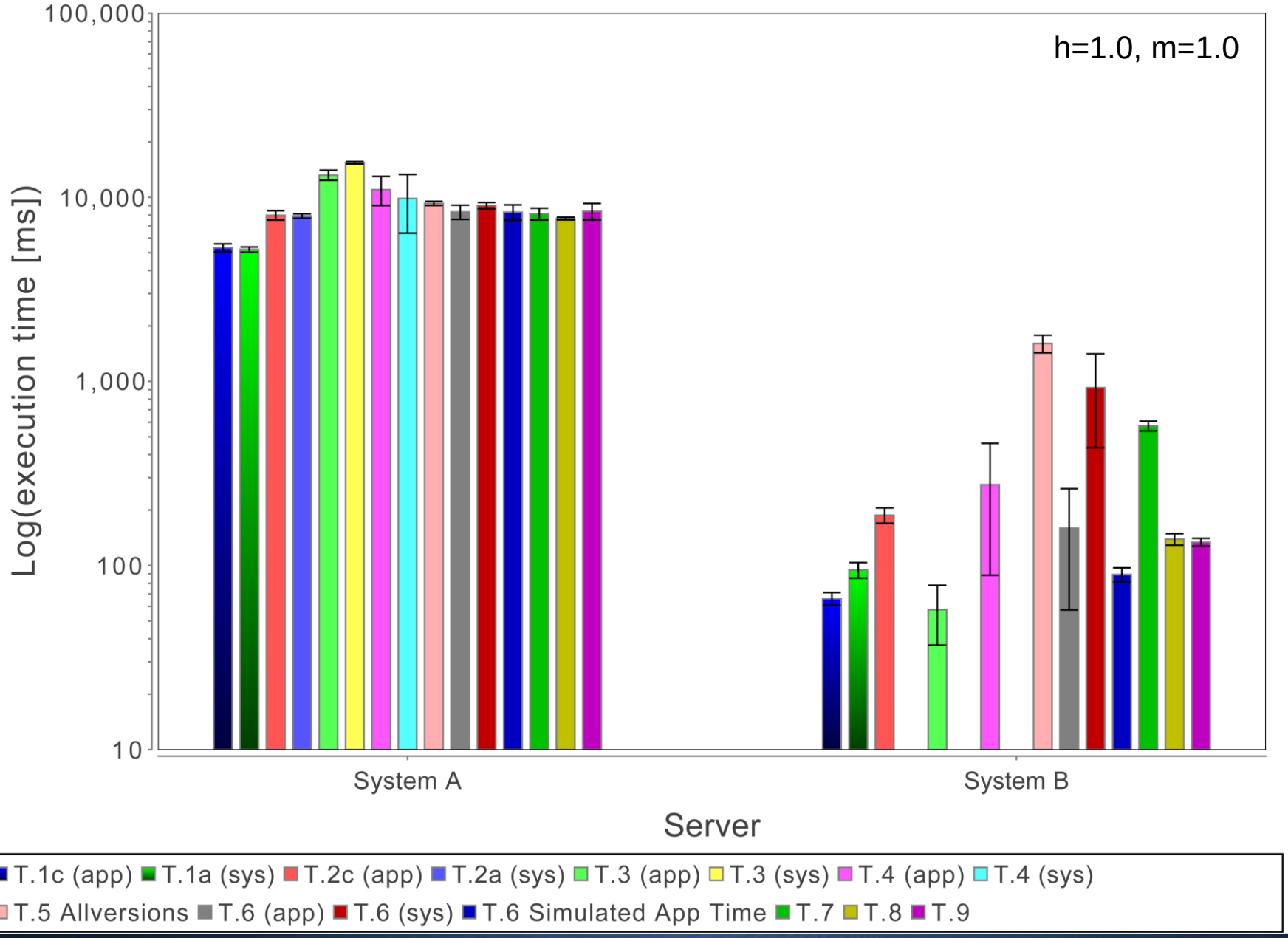
- Motivation
- State-of-the-art
- The TPC-BiH Benchmark
- **Preliminary Results**
- Conclusion

Evaluation Setup

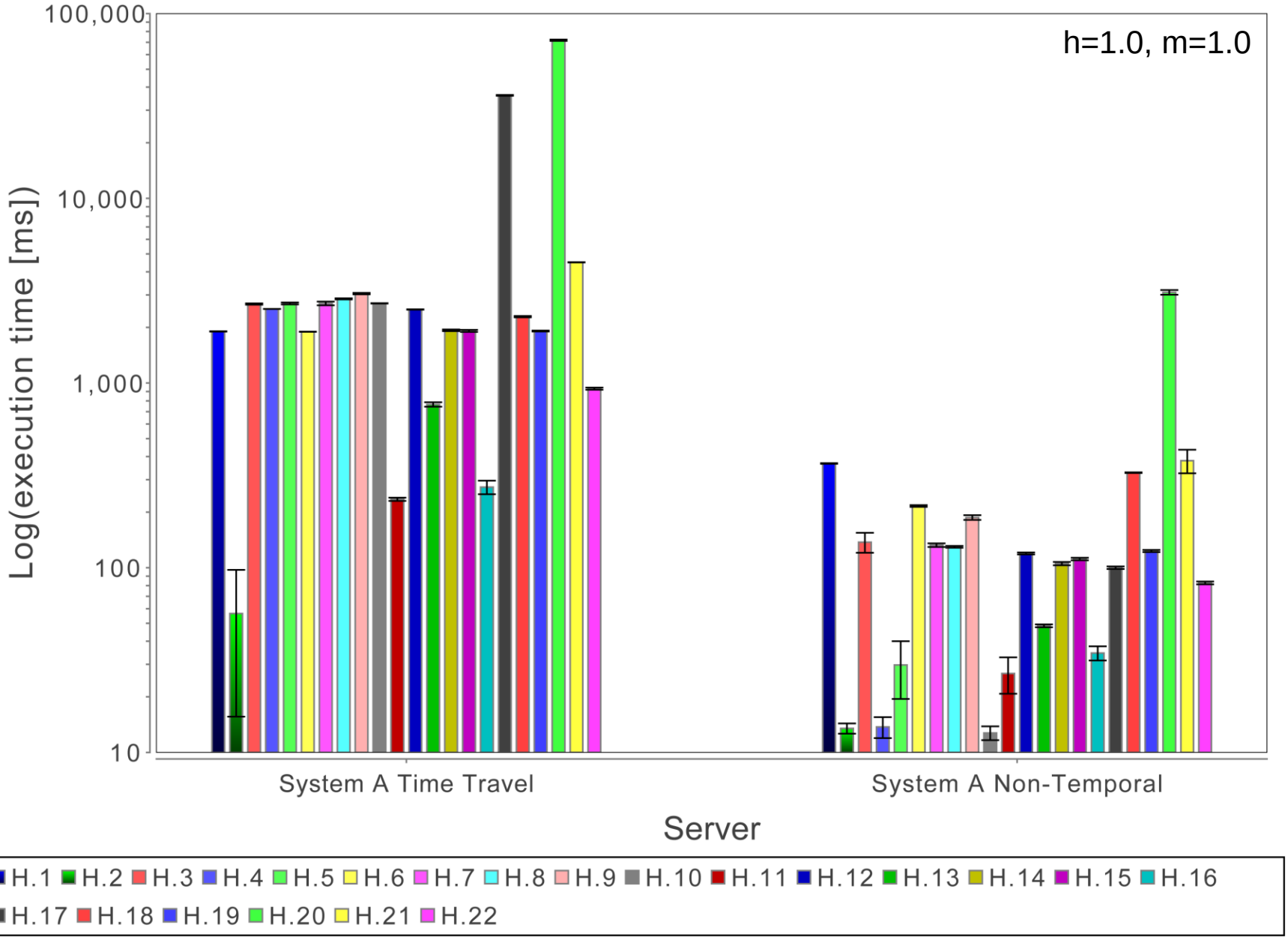
- Out-of-the-box settings (tuning is ongoing work)
- Two DBMS tested
  - System A: RDBMS supporting bitemporal workloads
  - System B: Main memory DBMS supporting system time
- Run on single system
  - 2 Intel Xeon X5675 processors with 6 cores at 3.06 GHz
  - 192GB of DDR3-1066MHz RAM
- Data
  - Initial data scaling factor (as in TPC-H): h=1.0
  - History size m=1.0 (1 million update scenarios)
  - Update scenario: Transaction (no bulk loading, but generate data!)
- At this stage purely performance results, no deep analysis

Note: logarithmic y-axis for all plots!

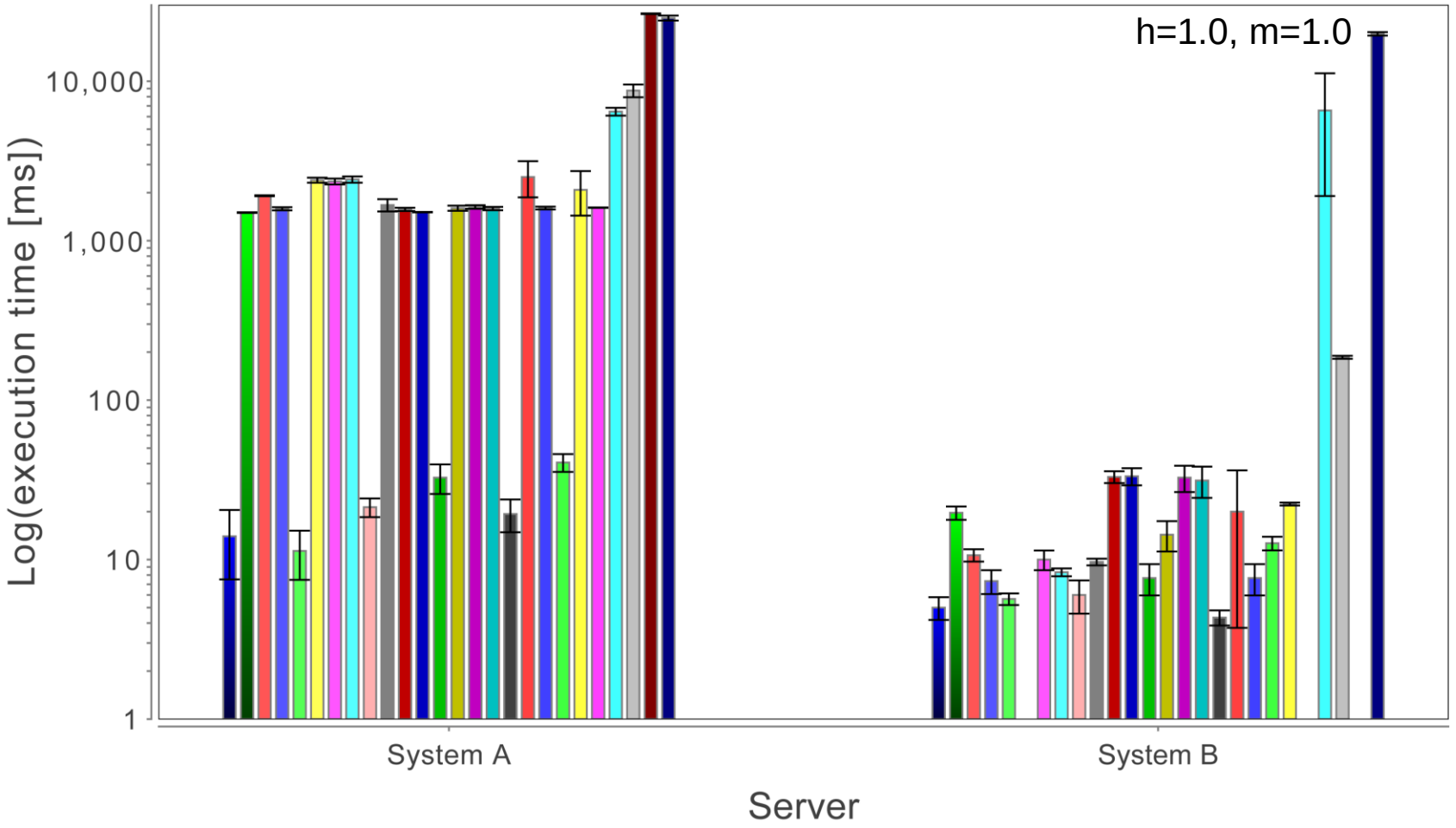
Pure Timeslice Queries



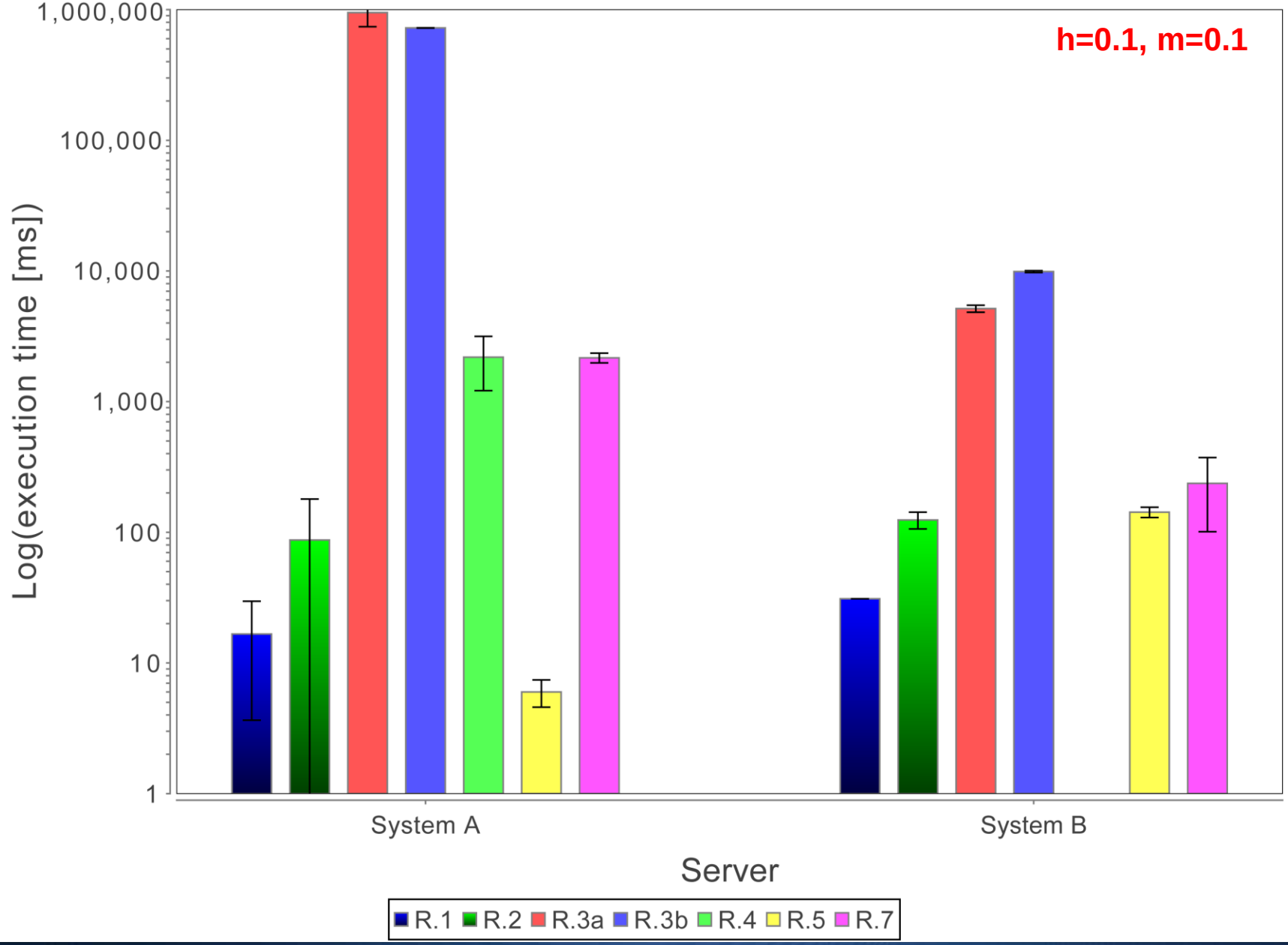
Normal vs. Bitemporal



Simple Aggregation over Time



Range-Timeslice Queries



Outline

- Motivation
- State-of-the-art
- The TPC-BiH Benchmark
- Preliminary Results
- **Conclusion**

Ongoing and Future Work

- Better coverage of DBMSs
- (Basic) tuning for temporal workloads, deeper analysis of query plans
- Variance in workloads
- Impact of scaling and value distribution
- Evaluation of update performance

TPC-BiH: Conclusion

Properties

- Builds on existing benchmarks
- Comprehensive coverage of use cases for temporal data
- Data generator based on real-world scenarios

Prelimilary Results for Commercial Systems

- Significant potential for optimization
- Not all common application use cases supported sufficiently

Backup Slides

Covering Bitemporal Dimensions