Indexing Bi-temporal Windows

Chang Ge[1], Martin Kaufmann[2], Lukasz Golab[1], Peter M. Fischer[3], Anil K. Goel[4]

1 UNIVERSITY OF WATERLOO

2 Systems@ETH zürich

3 UNI FREIBURG

4 SAP

Outline

Introduction

Bi-temporal Windows

Related Work

The BiSW Index

Experiments

Conclusion

Introduction

An example:

An ISP company provides Internet services with different plans.

On April 1, 2015, John signs a plan A.

The plan is activated on April 5. His 1-year service expires on April 5, 2016.

John switches to plan B on April 9, 2015

Question: how to model this story in database?

Introduction

An example:

An ISP company provides Internet services with different plans.

On April 1, 2015, John signs a plan A.

Customer	Plan	Sys Time
John	A	04-01-2015

Introduction

An example:

An ISP company provides Internet services with different plans.

On April 1, 2015, John signs a plan A.

The plan is activated on April 5. His 1-year service expires on April 5, 2016.

Customer	Plan	Sys Time	App Start	App End
John	A	04-01-2015	04-05-2015	04-05-2016

Introduction

An example:

An ISP company provides Internet services with different plans.

On April 1, 2015, John signs a plan A.

The plan is activated on April 5. His 1-year service expires on April 5, 2016.

John switches to plan B on April 9, 2015

Customer	Plan	Sys Time	App Start	App End
~~John~~	~~A~~	~~04-01-2015~~	~~04-05-2015~~	~~04-05-2016~~

Information is lost!

Introduction

An example:

An ISP company provides Internet services with different plans.

On April 1, 2015, John signs a plan A.

The plan is activated on April 5. His 1-year service expires on April 5, 2016.

John switches to plan B on April 9, 2015

Customer	Plan	Sys Start	Sys End	App Start	App End
John	A	04-01-2015	04-09-2015	04-05-2016	04-05-2016
John	B	04-09-2016	∞	04-05-2016	04-05-2016

Introduction

Application time interval

- the time period during which a fact is true from the view of real world.
- E.g. *“the 2013-year income” – perspective of facts*

System(transaction) time interval

- the time period during which a fact stored in the database is to be true.
- E.g*. “the 2013-year income recorded at 2014” – view of database*

Bi-temporal table

- A non-temporal table with one *application time interval* and one *system time interval*.

Bi-temporal data model

Bi-temporal table

Customer	Plan	Sys Start	Sys End	App Start	App End
John	A	100	110	10	13
John	B	110	∞	8	10
Bob	A	110	∞	20	∞

Lifetime interval for each time dimension.

Open-ended intervals describe undetermined end.

Time dimensions not necessarily correlated:

App Time *before* System Time: delay, correction of past

App Time *after* System Time: future action, forecast

Sometimes, not even comparable

Bi-temporal data model

Bi-temporal table

Customer	Plan	Sys Start	Sys End	App Start	App End
John	A	100	∞	10	13

System time updates are ordered and append-only.

Bi-temporal data model

Bi-temporal table

Customer	Plan	Sys Start	Sys End	App Start	App End
John	A	100	110	10	13
John	B	110	∞	8	15
Bob	A	110	∞	20	∞

System time updates are ordered and append-only.

Application time updates are arbitary, but trigger a new system version.

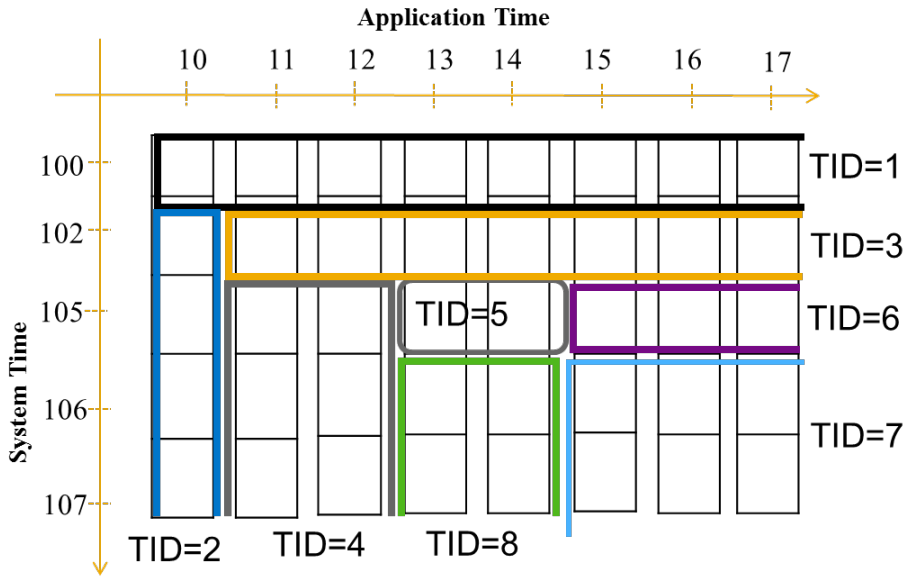
Times are not symmetric.

Bi-temporal data model

Bi-temporal table

ID	Balance	StartApp	EndApp	StartSys	EndSys
1	50	10	∞	100	102
2	50	10	11	102	∞
3	40	11	∞	102	105
4	30	11	13	105	∞
5	100	13	15	105	106
6	30	15	∞	105	106
7	35	15	∞	106	∞
8	90	13	15	106	∞

2D space



Introduction – window queries

Fixed bi-temporal window query:

“Return the customer plans for application period of Jan 1 to 30 as they were known between (system) time May 1 to 30”

One window fixed, slide the other:

“From January 1 till December 1, return the customer plans for the past 30 days, as they were known between December 1 and 30”

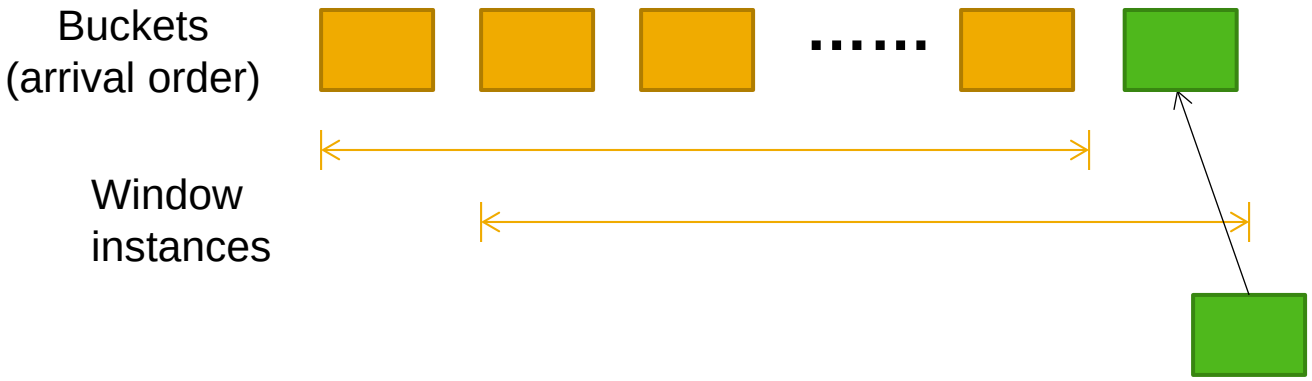
“At the end of each day, return the customer plans for the application time period of April 1 to 30 as they were known in the past 14 days”

Both windows slide:

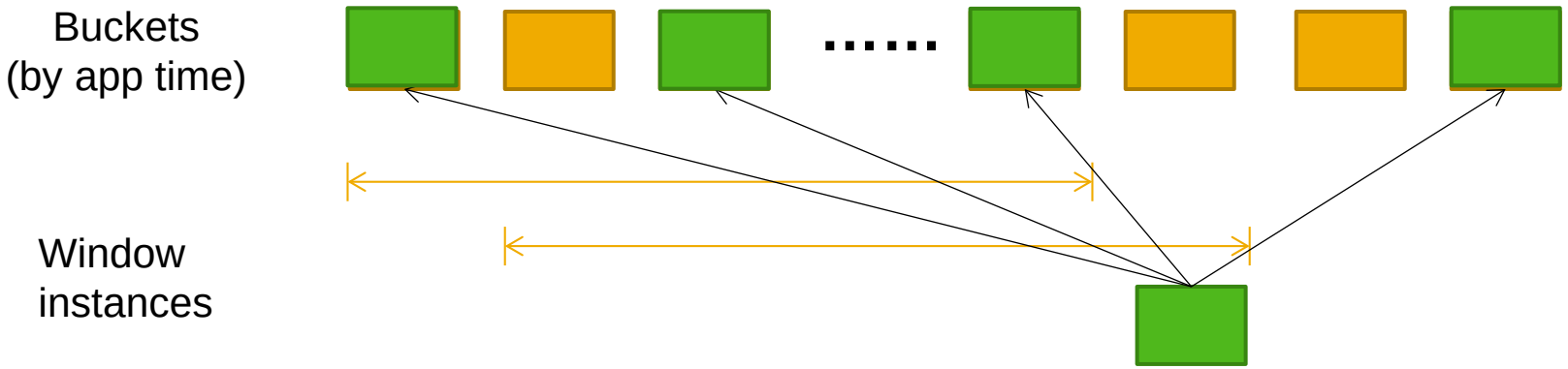
“Return the customer plans for the past 30 days as they were known over the past 14 days”

Introduction – sliding windows

Slide system time only



Slide application time only



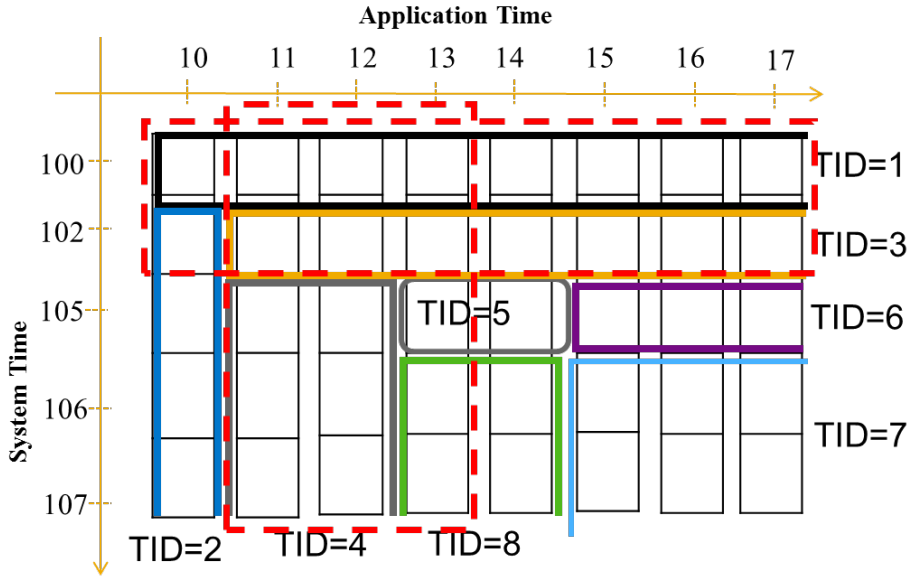
Bi-temporal Windows

Bi-temporal windows

Bi-temporal table

ID	Balance	StartApp	EndApp	StartSys	EndSys
1	50	10	∞	100	102
2	50	10	11	102	∞
3	40	11	∞	102	105
4	30	11	13	105	∞
5	100	13	15	105	106
6	30	15	∞	105	106
7	35	15	∞	106	∞
8	90	13	15	106	∞

Window over bi-temporal data



Two window semantics:

- **Interval oriented.**
- **Event oriented.**

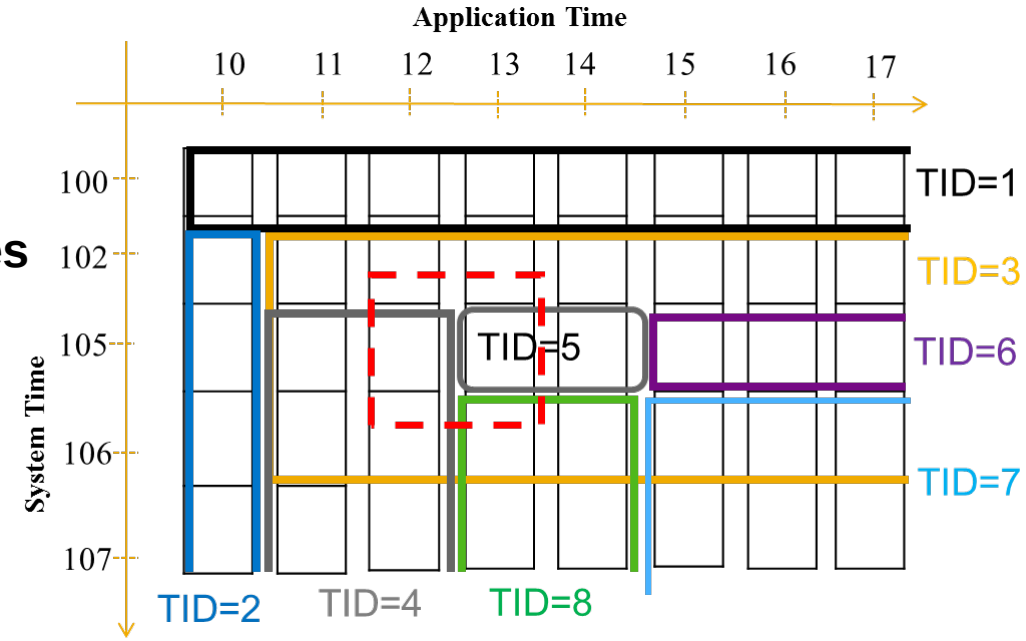
Bi-temporal windows

Interval oriented windows

- **Ranges overlap with window boundaries**
- **e.g. TID=3,4,5,8 in the window**

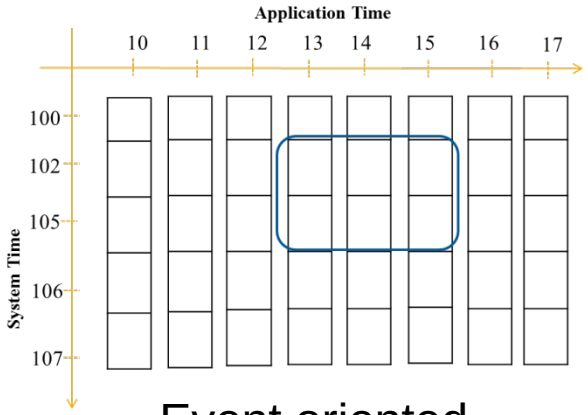
Event oriented windows

- **Ranges in the window boundaries**
- **e.g. TID=4,5,8 in the window**

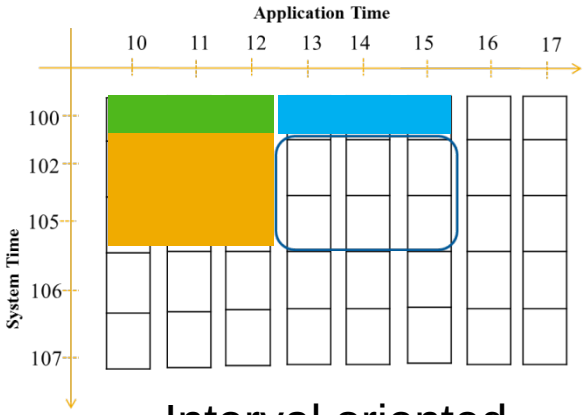


Bi-temporal window queries

Fixed window queries

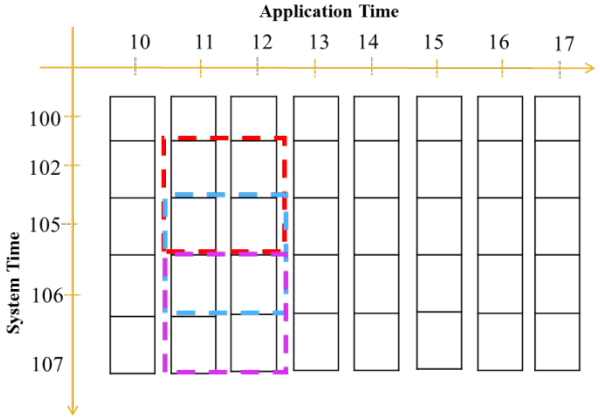


Event oriented

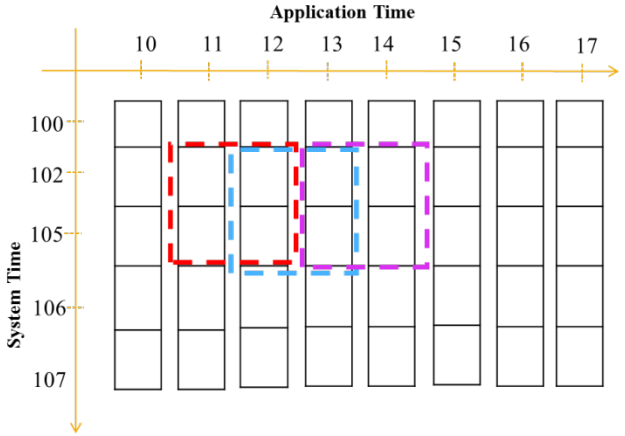


Interval oriented

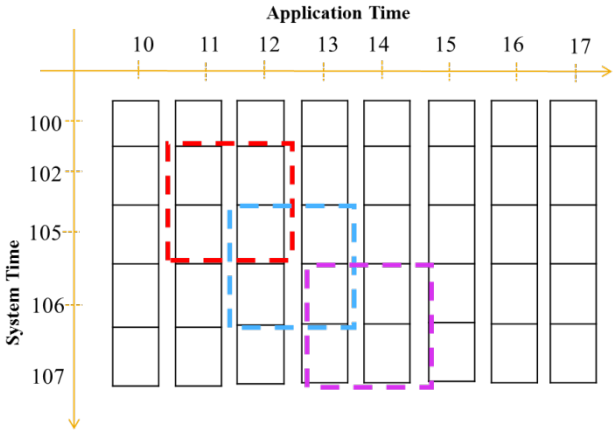
Slide on different time dimensions



Slide system time



Slide application time



Slide both times

Problem statement

Given a (static or continuously updating) bi-temporal table, we want to support:

- **Fixed bi-temporal windows.**
- **3 cases of sliding windows, both interval and event oriented, and both over streaming and historical data.**

To be specific,

- **Fast insert.**
- **Efficient indexing dimensions.**
- **Incremental computation over sliding windows.**

Related Work

Related work – commercial DBMS

DB2 supports bitemporal table, but no temporal sliding window support.

Oracle Flashback Data Archive supports limited system time queries.

Teradata functionally rewrites temporal query by adding time-based constrains.

SAP HANA supports temporal operators on system time.

Related work – system time sliding window

Existing data stream systems do not support bitemporal data model

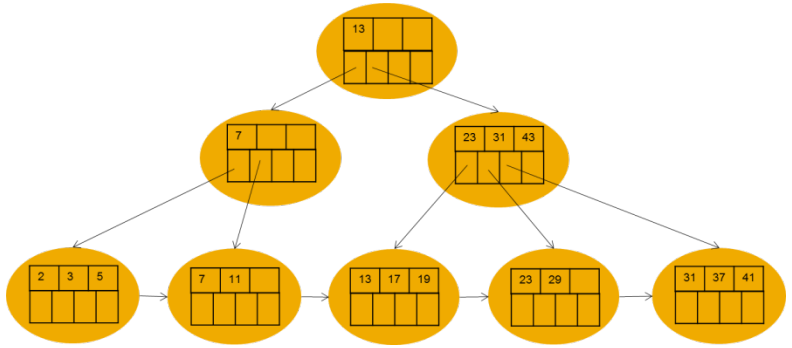
Most of those assumes system time ordering

Application time support is limited as exceptions

Related work – spatial-temporal trees

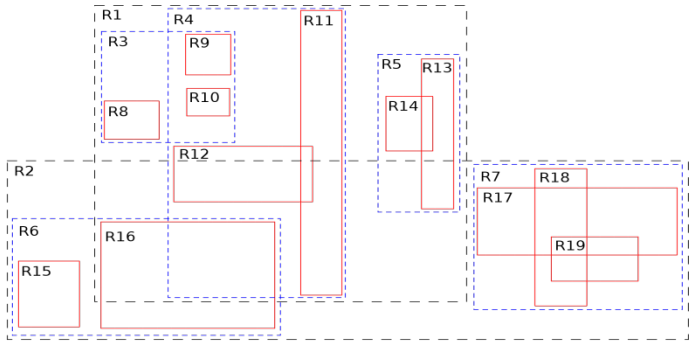
Alternatives

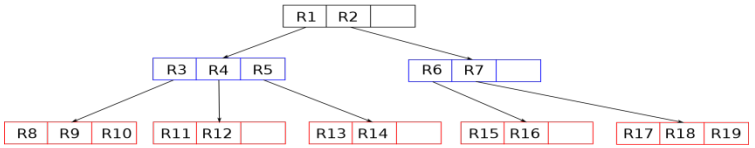
- **B-trees**
- **R-trees**



Drawbacks

- **High cost on maintenance**
- **Open intervals are not efficient**

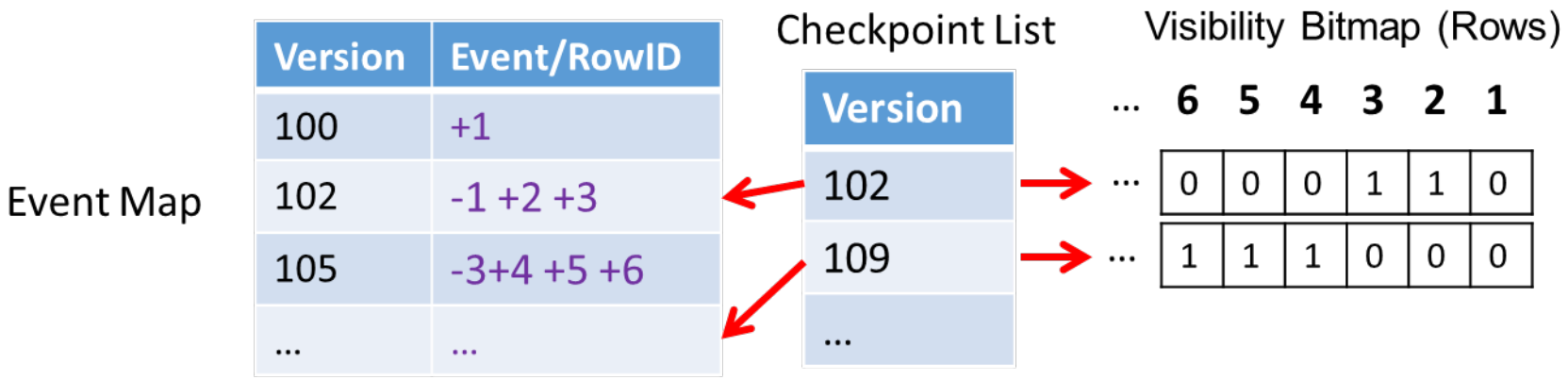




Related work – Timeline Index[1]

Timeline Index

- **Log-oriented index over changed rows.**
- **Good performance for queries and maintenance**

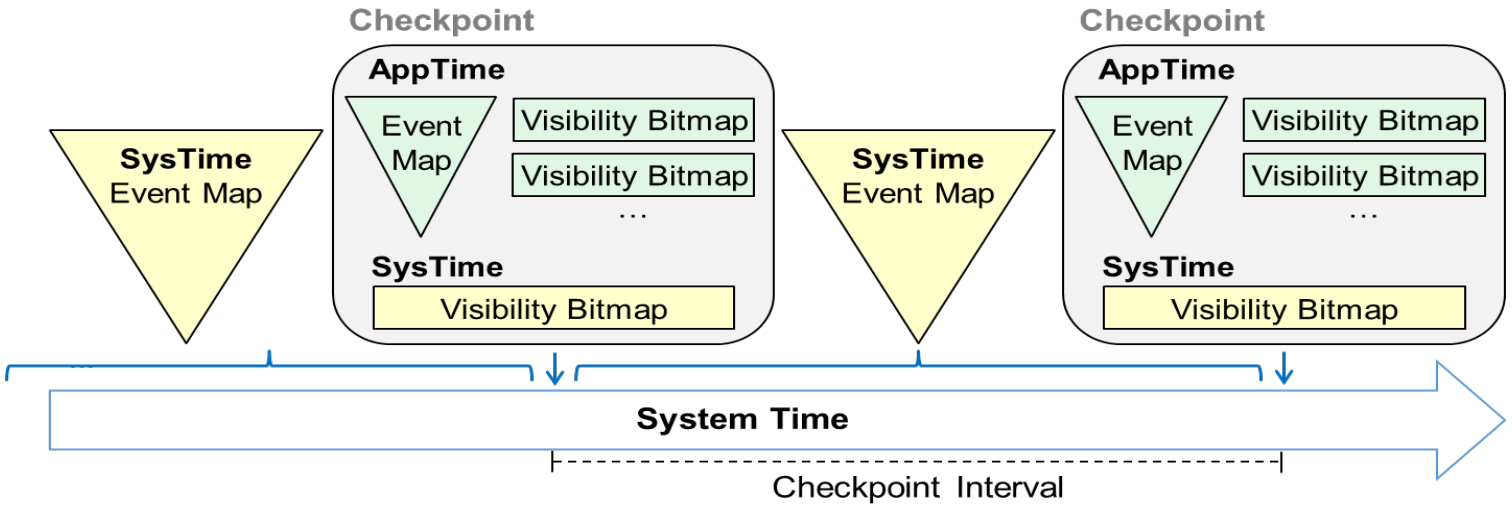


[1] "*Timeline Index: A Unified Data Structure for Processing Queries on Temporal Data in SAP HANA,*" SIGMOD 2013

Related work – Bi-temporal Timeline Index[2]

BiTL

- **Timeline Index for both dimensions**
- **Lazy materialization on application time**



[2] "*Bi-temporal timeline index: A data structure for processing queries on bi-temporal data,*" ICDE 2015

The BiSW Index

Introduction

Bi-temporal Windows

Related Work

The BiSW Index

- Logical design
- Physical design
- Query processing

Experiments

Conclusion

The BiSW Index – logical design

Two-dimensional grid

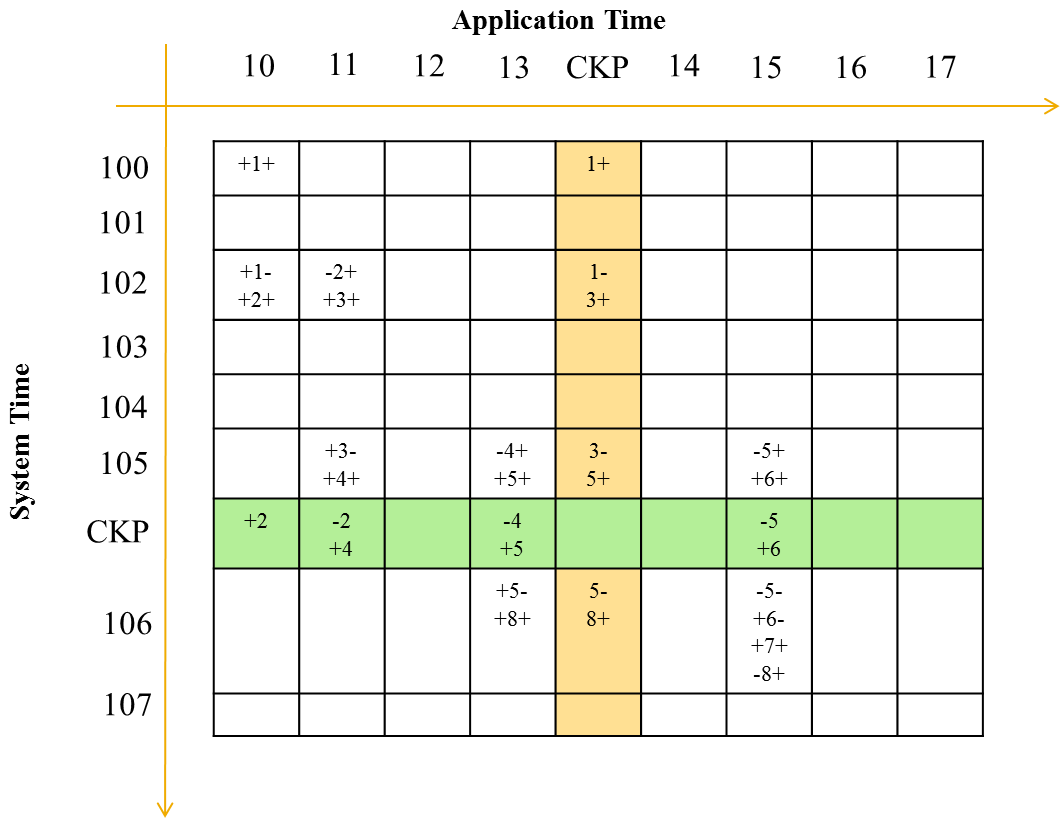
Event grid

Events have roles

- $opAppTIDopSys$

Checkpoints

ID	Balance	StartApp	EndApp	StartSys	EndSys
1	50	10	∞	100	102
2	50	10	11	102	∞
3	40	11	∞	102	105
4	30	11	13	105	∞
5	100	13	15	105	106
6	30	15	∞	105	106
7	35	15	∞	106	∞
8	90	13	15	106	∞

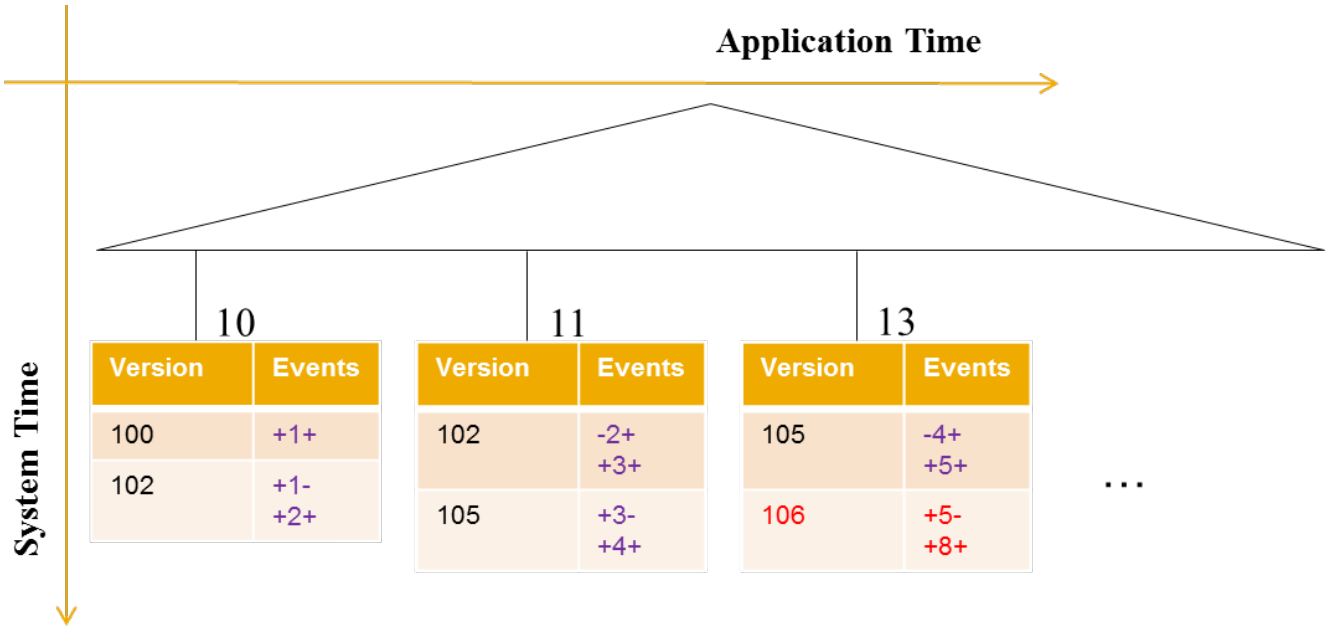


The BiSW Index – physical design

Application time partitioned system timeline index

Append-only

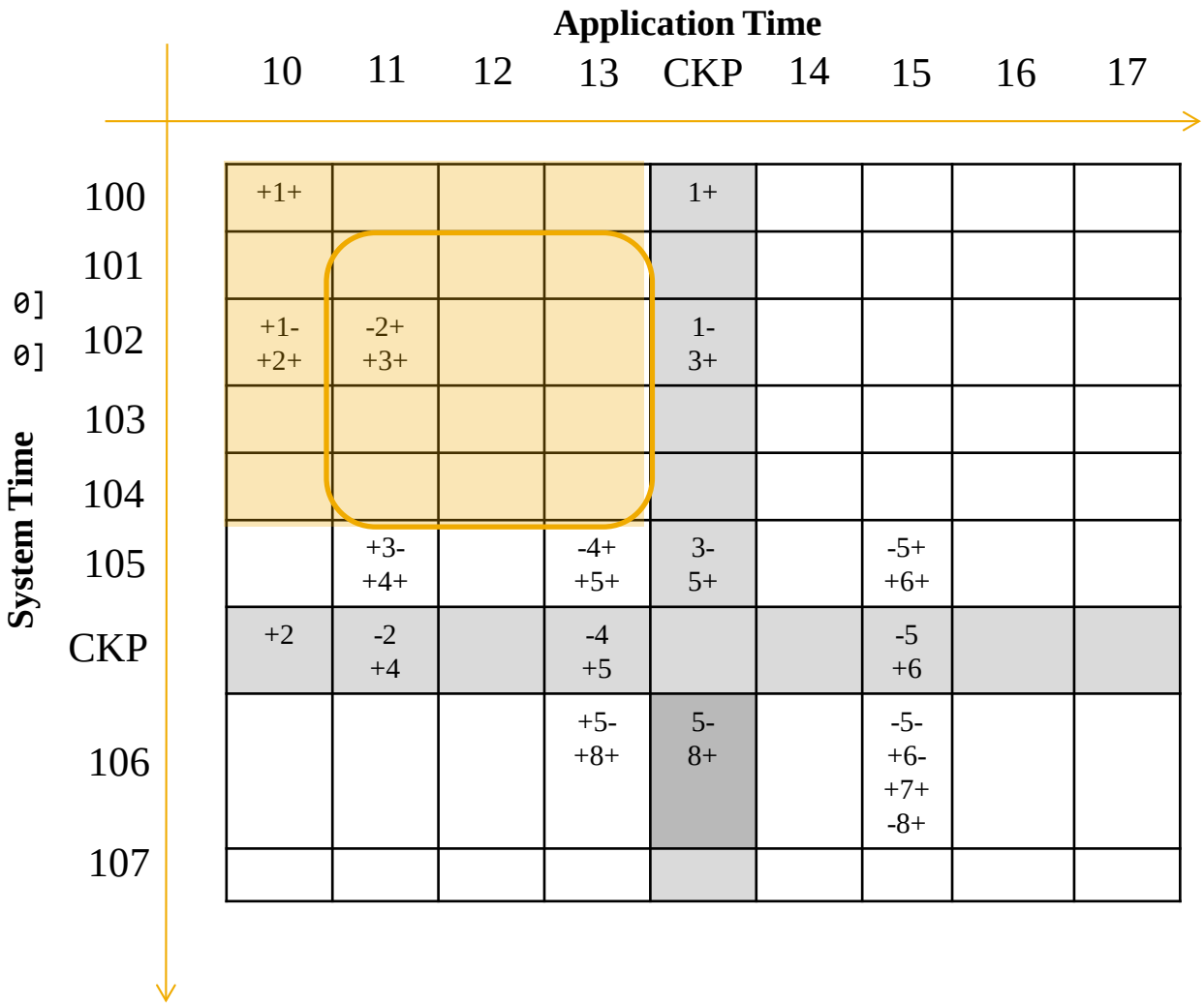
Decouple application time from system time



The BiSW Index – query processing

Fixed window

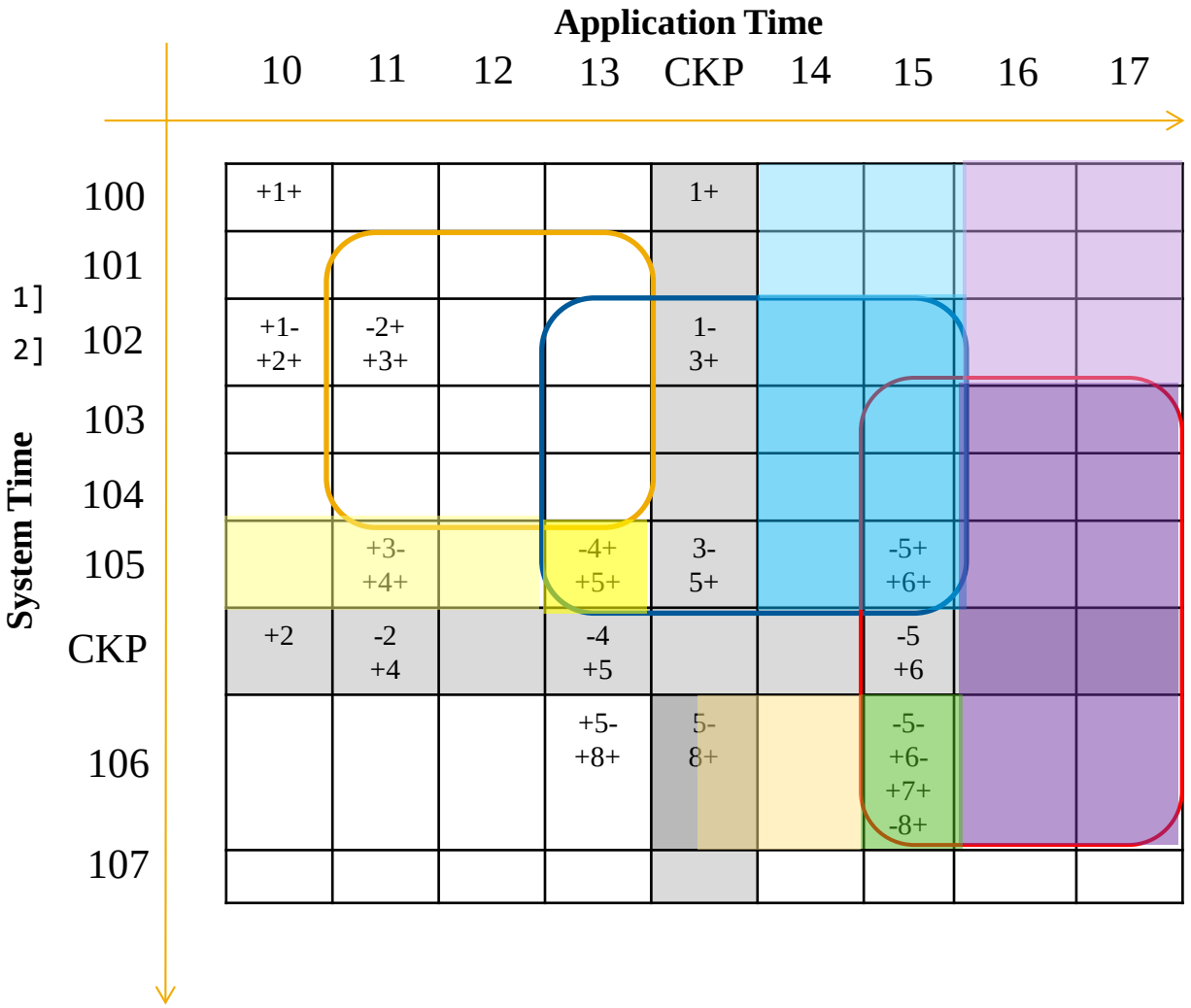
```
select *
from table
    [SYS, START 101, RANGE 3, SLIDE 0]
    [APP, START 11,  RANGE 2, SLIDE 0]
    [RATIO 1:1]
```



The BiSW Index – query processing

Sliding window

```
select *
from table
    [SYS, START 101, RANGE 3, SLIDE 1]
    [APP, START 11,  RANGE 2, SLIDE 2]
    [RATIO 1:1]
```



Experiments

Experiments

Comparisons

- **BiTL**
- **B+-Tree**
- **RR*-Tree**
- **Table scan**

Test environment

- **16 core, 256GB Linux box, gcc 4.7**
- **Indices and table columns are memory resident**

Experiments

Data: TPC-BiH benchmark

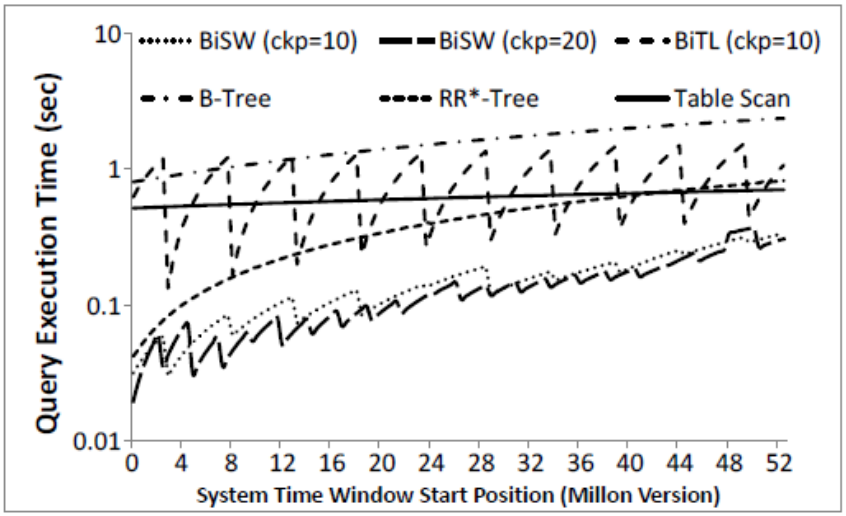
Data Set	SF_0	SF_H	#tuple	#app.version	#sys.version
1	1	10	3M	3M	3M
5	1	50	17M	15M	15M
10	1	100	57M	55M	55M

Measurements

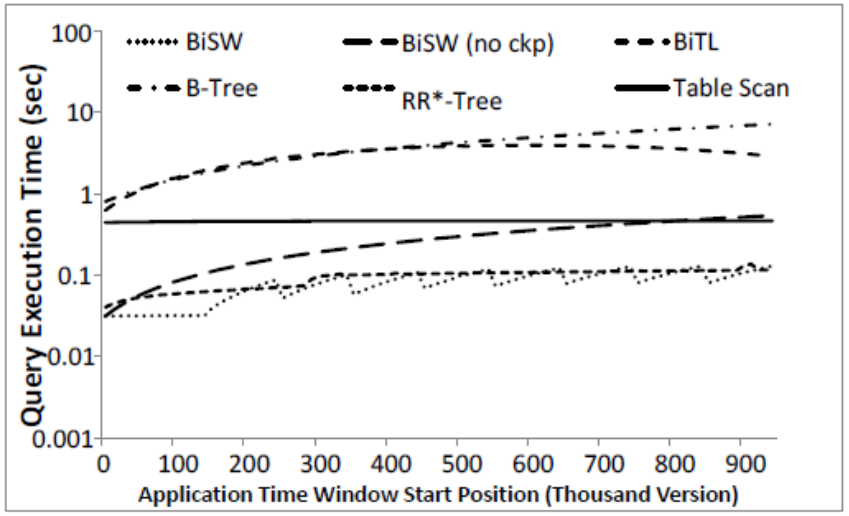
Experiment	Measurement
1	Fixed window queries
2	Sliding window on system time
3	Sliding window on application time
4	Sliding on both times
5	Data skew
6	Space and maintenance
7	Continuous queries

Experiments

Fixed interval oriented window as a function of the starting position of the window



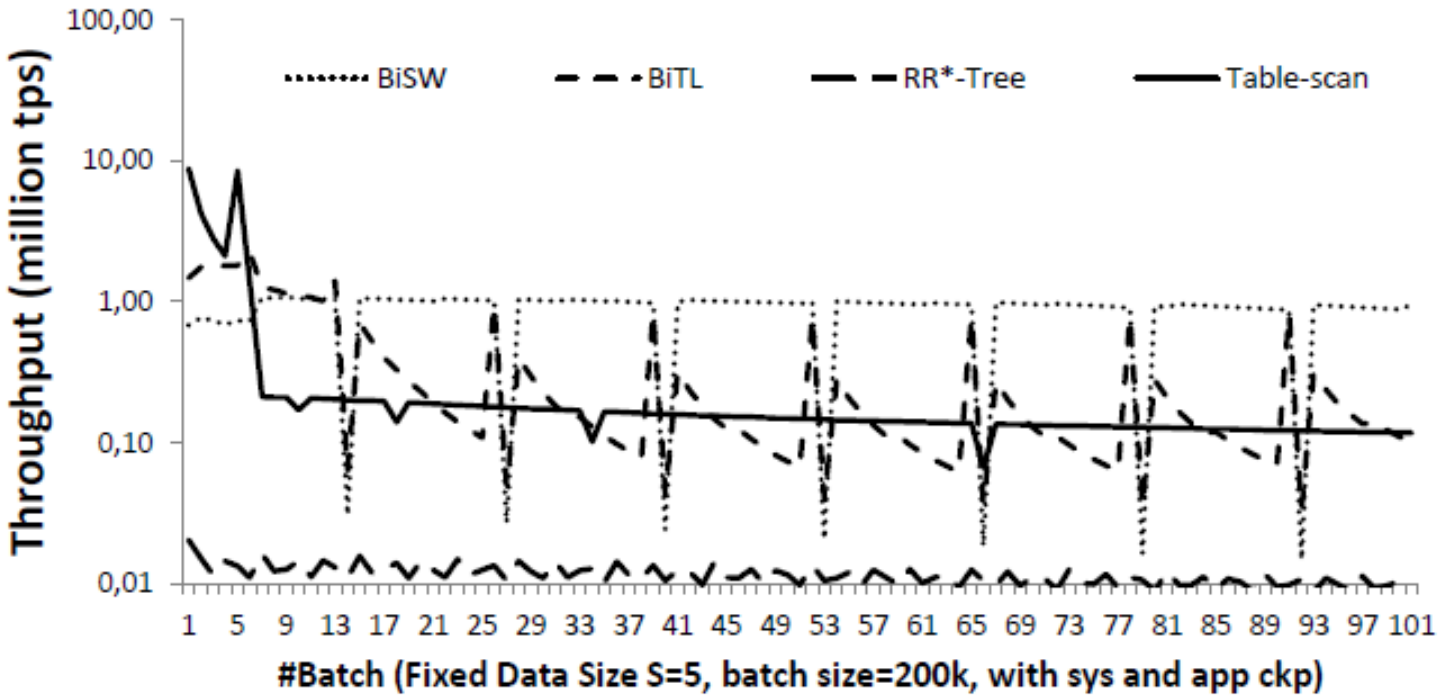
(a) System Time



(b) Application Time

Experiments

Slide both times over continuous queries



Experiments – lessons learned

BiSW wins in query performance, often by a significant margin.

BiSW consistently outperforms all competitors for continuous queries.

BiSW treats space and maintenance of checkpoints for better performance, but the overhead is small.

Temporal skew has a major compact, but BiSW is very tolerant.

Conclusion

Introduction

Bi-temporal Windows

Related Work

The BiSW Index

Experiments

Conclusion

Conclusion

This work initiated a study of bi-temporal windows, which bring together two important concepts of bi-temporal database and window queries over temporal attributes.

We proposed and formulated an index structure to support bi-temporal window queries.

BiSW was intensively evaluated against state-of-the-art approaches and won on query performance, space overhead and maintenance cost.

Open problems

- **Query optimization**
- **Benchmarking**