GraphQ 2017

EDBT ICDT

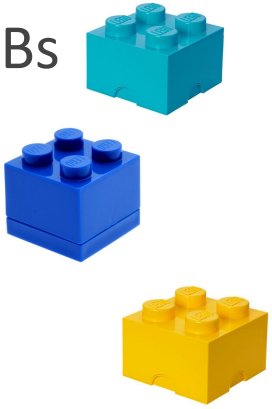
OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG

INF

DBSE

Backlogs & Interval Timestamps:

Building blocks for supporting temporal queries in graph DBs



Gabriel Campero Durand, Marcus Pinnecke, David Broneske, Gunter Saake

Work in progress paper@GraphQ Workshop 2017

Temperance: The virtue of keeping track of time

1439-1442

Porta della Carta (Between St. Mark's Church and the Ducal Palace)

Maestro Bortolo Tajapietra

1 Temporal models in data mgmt

A plethora of temporal models exists...

Point-Stamped Temporal Models

David Toman
University of Waterloo, Waterloo, ON, Canada

Synonyms
Point-based temporal models

Period-Stamped Temporal Models

Nikos A. Lorentzos
Agricultural University of Athens, Athens, Greece

Synonyms
Interval-based temporal models

Telic Distinction in Temporal Databases

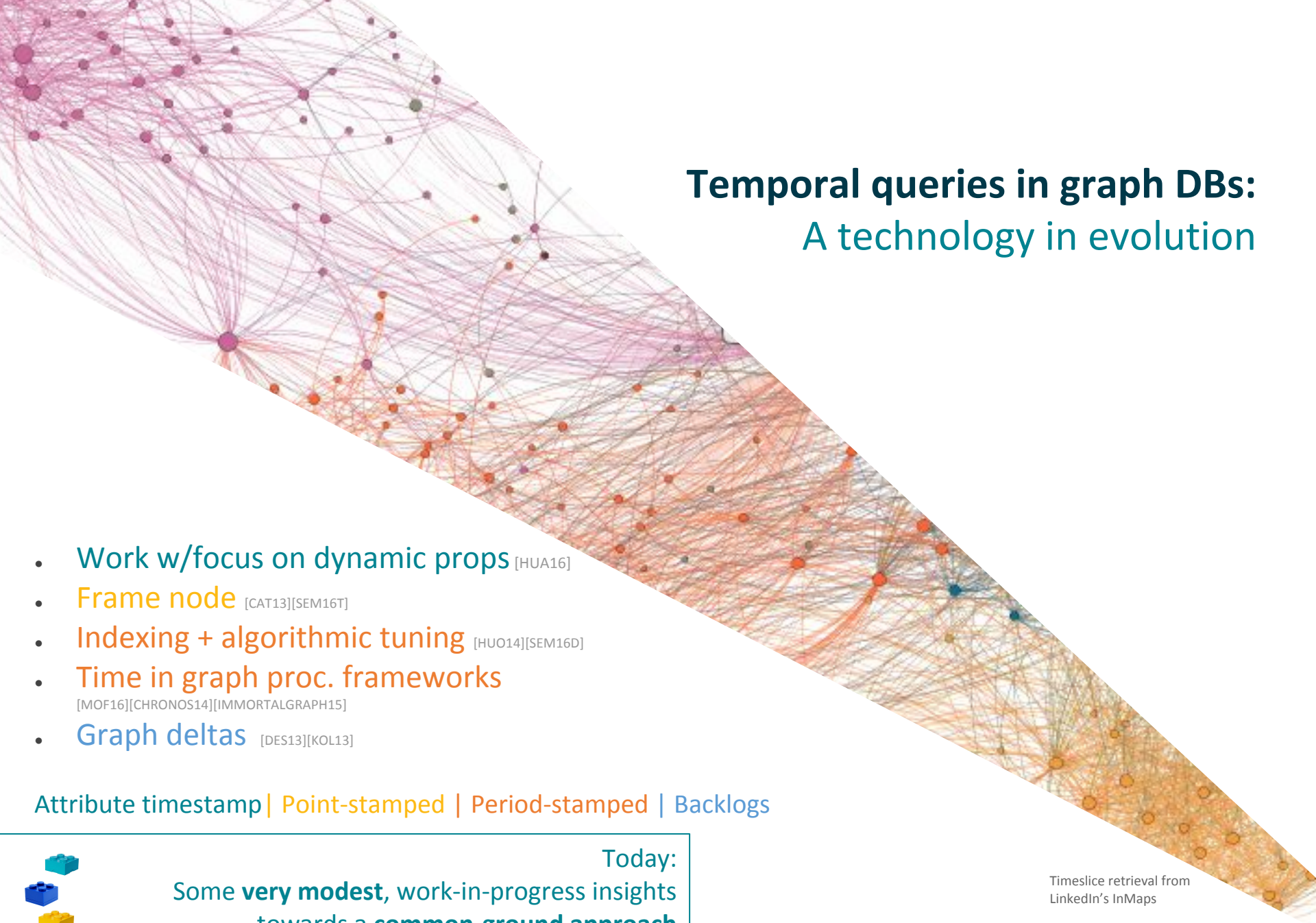
Vijay Khatri[1], Richard T. Snodgrass[2], Paolo Terenziani[3]
[1]Indiana University, Bloomington, IN, USA
[2]University of Arizona, Tucson, AZ, USA
[3]University of Turin, Turin, Italy

Synonyms
Point-versus period-based semantics

They help to express clearly:

- the temporal validity of items
- the sequence of changes on the items

Temporal queries in graph DBs:
A technology in evolution



- Work w/focus on dynamic props [HUA16]
- Frame node [CAT13][SEM16T]
- Indexing + algorithmic tuning [HUO14][SEM16D]
- Time in graph proc. frameworks [MOF16][CHRONOS14][IMMORTALGRAPH15]
- Graph deltas [DES13][KOL13]

Attribute timestamp | Point-stamped | Period-stamped | Backlogs

Today:
Some **very modest**, work-in-progress insights
towards a **common-ground approach**

Timeslice retrieval from
LinkedIn's InMaps

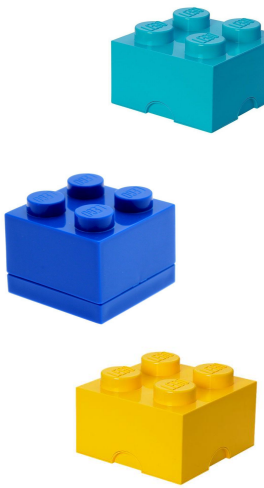
3. Towards adaptive temporal views

2. Query rewriting to use views
(Snapshot -> Differential processing)

1. Building blocks for temporal views

1 Design choices related to time support

- Separating snapshots, aggregation to a **single graph**, **or hybrids**.
- Proposal #1: (atelic) validity can be fully captured with **interval timestamps**.
  - More expressive than point timestamps, but special care required for telic case.
- After adopting interval timestamps as a model, other choices:
  - **One interval of validity per item***, or several.
  - Tracking attribute or **topology changes**.



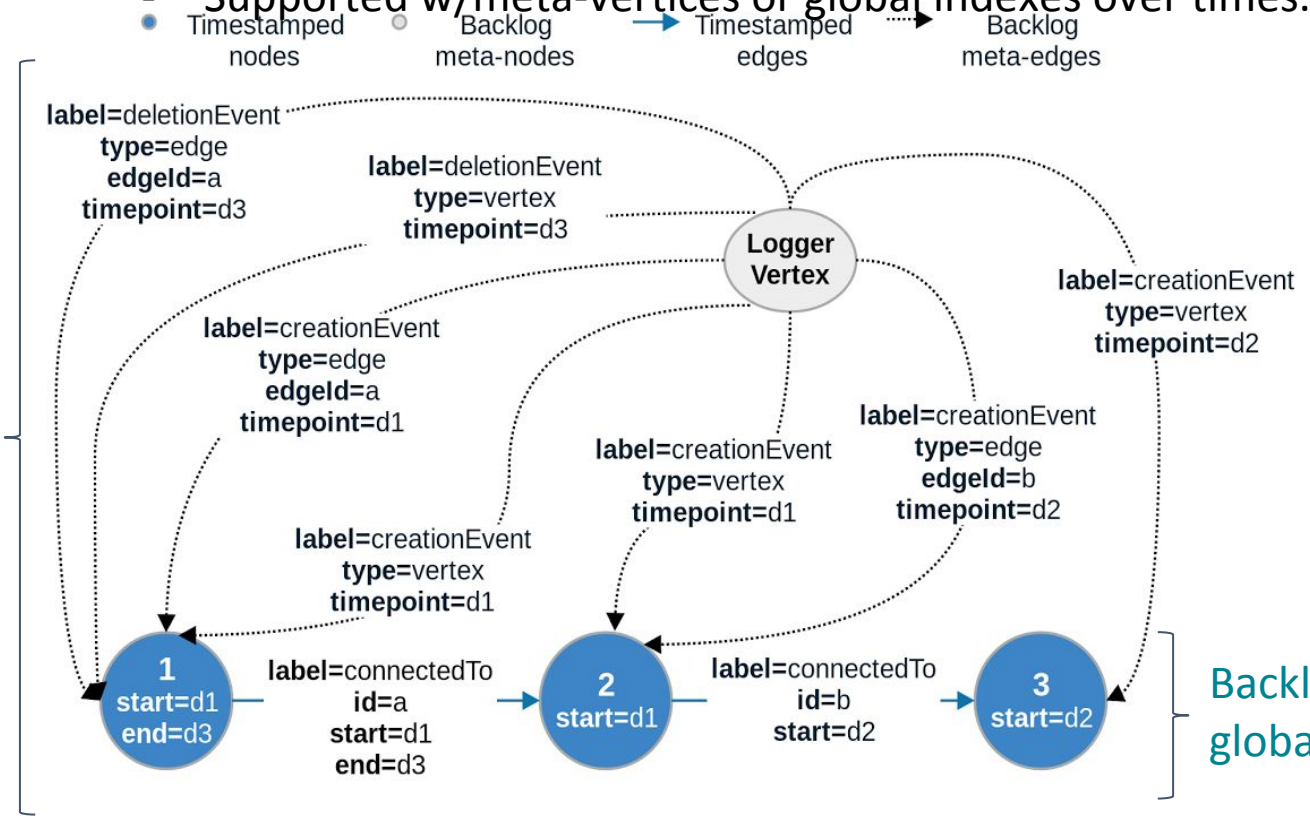
In **bold**, the choices for our study.

*Arguably easier to index.

1 Backlogs and Interval Timestamps

Proposal #2: backlogs can be understood as **an emergent "view"**, which comes for free by not having all snapshots separate.

- **Differential processing**.
- Supported w/meta-vertices or global indexes over times.



Backlogs with meta-items (Using the graph as an index, GRAIN)

Backlogs with global indexes

1 Backlogs and Interval Timestamps

Note:

- The concept of using the graph itself as an index, to create certain “views”, builds on the possibilities for combining models with meta-models in graph databases.
- We believe that this is a powerful aspect of these databases & that it can be crucial to adaptive features (e.g. cracking views to accelerate graph traversals).
- This paper provides an early example of opportunities from this technique.

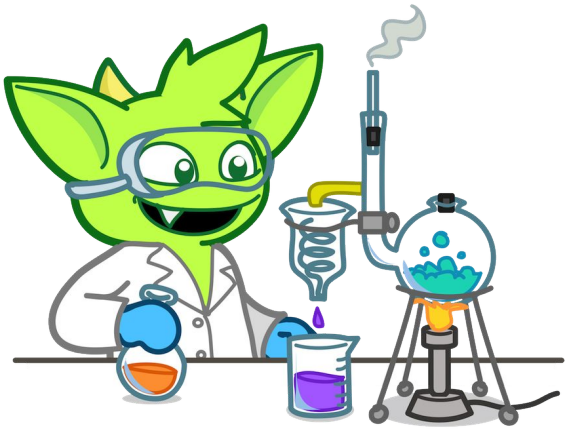

2. Query rewriting to use views
(Snapshot -> Differential processing)
1. Building blocks for temporal views

2 Density Calculation

Density : $$\frac{|E|}{|V|\,(|V|-1)}$$

Basic query:

```
1  titanGraph.V().
2      hasLabel("device").
3      has("deletedAt", P.gt(date)).
4      has("createdAt", P.lte(date)).
5      count().next();
```

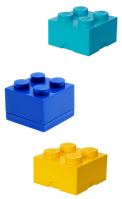


Incremental- Using global indexes:

```
1  titanGraph.V().
2      hasLabel("device").
3      has("createdAt", date).
4      count().next();
5  titanGraph.V().
6      hasLabel("device").
7      has("deletedAt", date).
8      count().next();
```

Incremental- GRAIN:

```
1  Map<Object,Long> creationCount =
2  (Map<Object, Long>) titanGraph.V(logId).
3      outE("created").
4      has("createdAt", date).
5      groupCount("created").by("createdType").
6      cap("created").next();
7  numCreatedVertices= creationCount.get(Vertex.class);
8  numCreatedEdges= creationCount.get(Edge.class);
```



2 Observed Performance

Figure 4: Avg. response time of density calculation

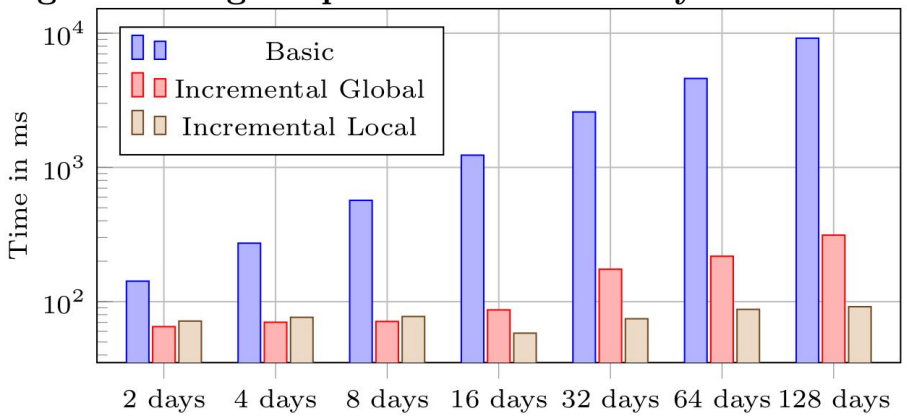
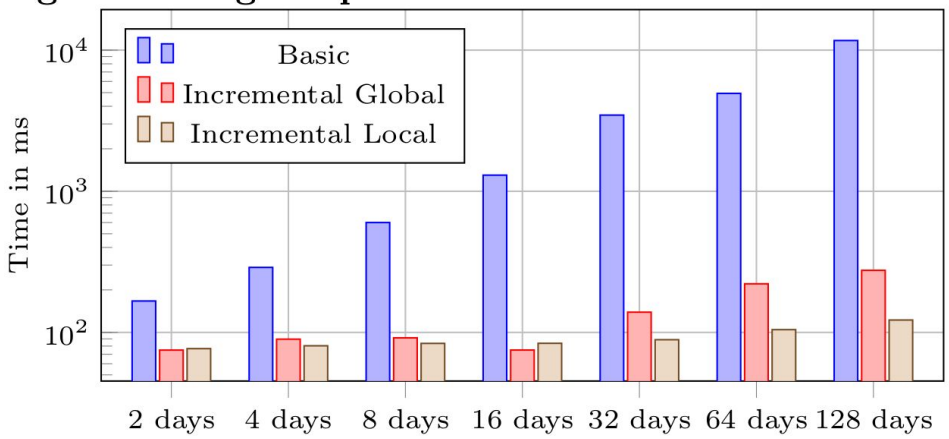


Figure 5: Avg. response time of time slice retrieval



SNAP as-733 dataset.

From 2-100x speedups by incremental processing.

The global was more heap intensive than the local (GRAIN), which was compute-bound.

The gains from incremental processing generalize to other methods.

TITAN

3. Towards adaptive temporal views

2. Query rewriting to use views
(Snapshot -> Differential processing)

1. Building blocks for temporal views

3 Towards adaptive temporal views

After establishing the use of interval timestamps, one-per-item.

The knobs we (as data managers) see:

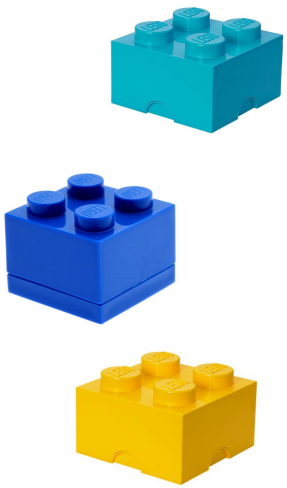
- Aggregated vs. Snapshot-by-snapshot (SBS)
- Incremental vs. SBS processing
- Different approaches for backlogs (room for improvement here).

What we need to figure out about using these knobs:

- Trade-off between the detrimental effect of structural locality loss vs. the opportunities from incremental processing in the aggregated version?
  - How to gauge this on a live system?
  - Effect of the different backlog approaches?
- On live systems, will SBS be better for new data, whereas aggregated copies could be for past (read-only) data?
- Agreed-upon benchmarks and relevant use cases.

Our goal: Using these knobs **adaptively**:

- What are the best triggers and methods to go from SBS to aggregated?
- How to achieve automatic query re-writing from SBS to the best incremental processing?
- Extensions for attributes?



This research is part of our work on an evolving H^{3}TAP database: Mondrian


OTTO VON GUERICKE
UNIVERSITÄT
MAGDEBURG
INF
D B
S E
Questions? Ideas?
campero@ovgu.de
Thanks for
the attention :)
Special Thanks
to the Organizers:
GraphQ 2017
Slides Template:
Copyright © Showeet.com
showeet

Extra slides

References

- [CAT13] Cattuto, Ciro, Marco Quaggiotto, André Panisson, and Alex Averbuch. "Time-varying social networks in a graph database: a Neo4j use case." In *First International Workshop on Graph Data Management Experiences and Systems*, p. 11. ACM, 2013.
- [DES13] Khurana, Udayan, and Amol Deshpande. "Efficient snapshot retrieval over historical graph data." In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, pp. 997-1008. IEEE, 2013.
- [CHRONOS14] Han, Wentao, Youshan Miao, Kaiwei Li, Ming Wu, Fan Yang, Lidong Zhou, Vijayan Prabhakaran, Wenguang Chen, and Enhong Chen. "Chronos: a graph engine for temporal graph analysis." In *Proceedings of the Ninth European Conference on Computer Systems*, p. 1. ACM, 2014.
- [HUA16] Huang, Haixing, Jinghe Song, Xuelian Lin, Shuai Ma, and Jinpeng Huai. "TGraph: A Temporal Graph Data Management System." In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pp. 2469-2472. ACM, 2016.
- [HUO14] Huo, Wenyu, and Vassilis J. Tsotras. "Efficient temporal shortest path queries on evolving social graphs." In *Proceedings of the 26th International Conference on Scientific and Statistical Database Management*, p. 38. ACM, 2014.
- [KOL13] Koloniari, Georgia, Dimitris Souravlias, and Evaggelia Pitoura. "On graph deltas for historical queries." *arXiv preprint arXiv:1302.5549* (2013).
- [IMMORTALGRAPH15] Miao, Youshan, Wentao Han, Kaiwei Li, Ming Wu, Fan Yang, Lidong Zhou, Vijayan Prabhakaran, Enhong Chen, and Wenguang Chen. "ImmortalGraph: A system for storage and analysis of temporal graphs." *ACM Transactions on Storage (TOS)* 11, no. 3 (2015): 14.
- [MOF16] Moffitt, Vera Zaychik, and Julia Stoyanovich. "Towards a distributed infrastructure for evolving graph analytics." In *Proceedings of the 25th International Conference Companion on World Wide Web*, pp. 843-848. International World Wide Web Conferences Steering Committee, 2016.
- [SEM16T] Semertzidis, Konstantinos, and Evaggelia Pitoura. "Time Traveling in Graphs using a Graph Database." In *EDBT/ICDT Workshops*. 2016.
- [SEM16D]Semertzidis, Konstantinos, and Evaggelia Pitoura. "Durable graph pattern queries on historical graphs." In *Data Engineering (ICDE), 2016 IEEE 32nd International Conference on*, pp. 541-552. IEEE, 2016.