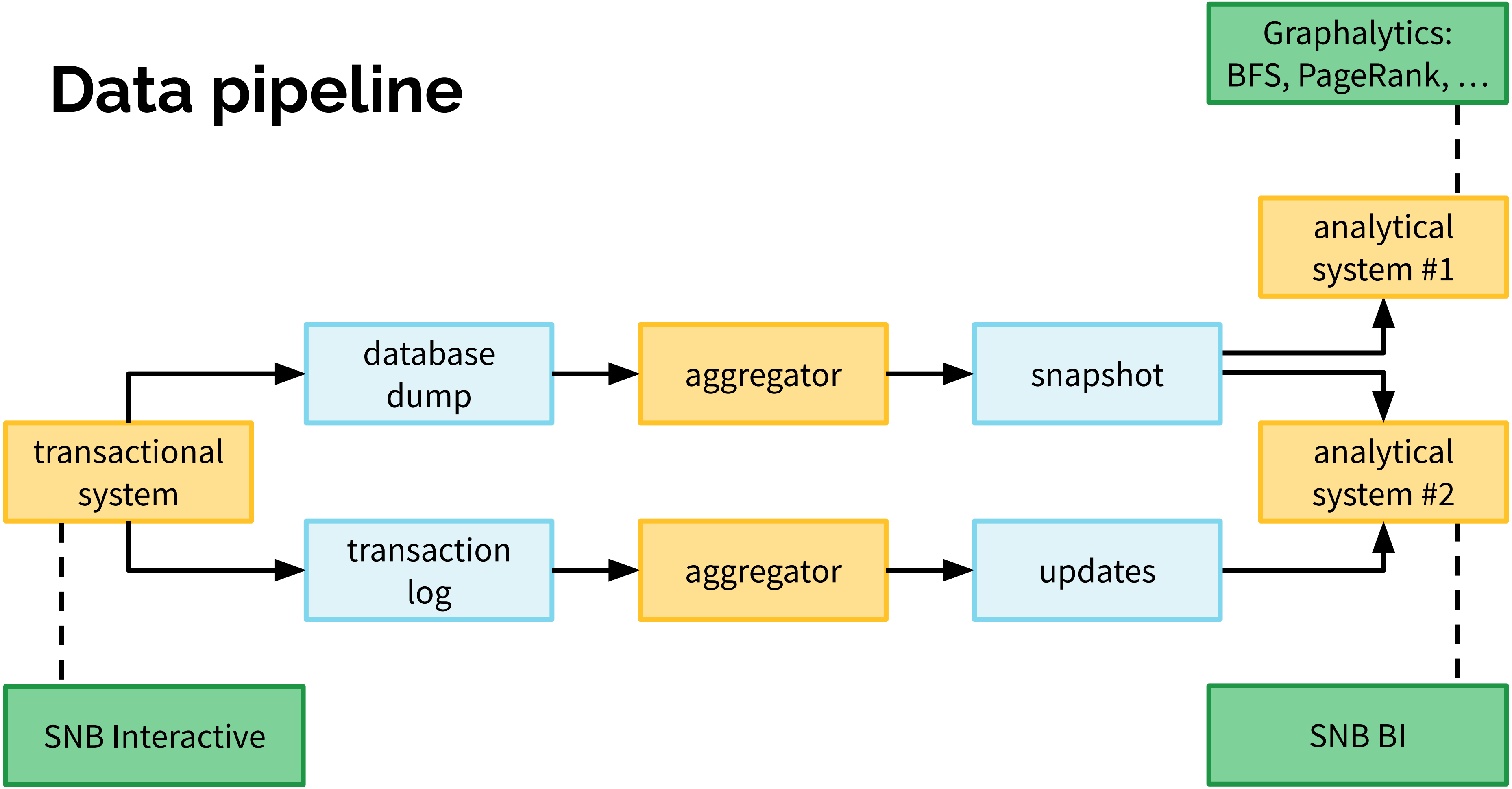
The LDBC Social Network Benchmark: Business Intelligence workload

Gábor Szárnyas
CWI

15th TUC meeting

Data pipeline



LDBC SNB BI workload

A modern OLAP benchmark suite

- Correlated, temporal graph data set
- Analytical queries, including graph operations
- Inserts & deep deletes
- Parameter curation

Social network data set

- Correlated
- Temporal

Example graph

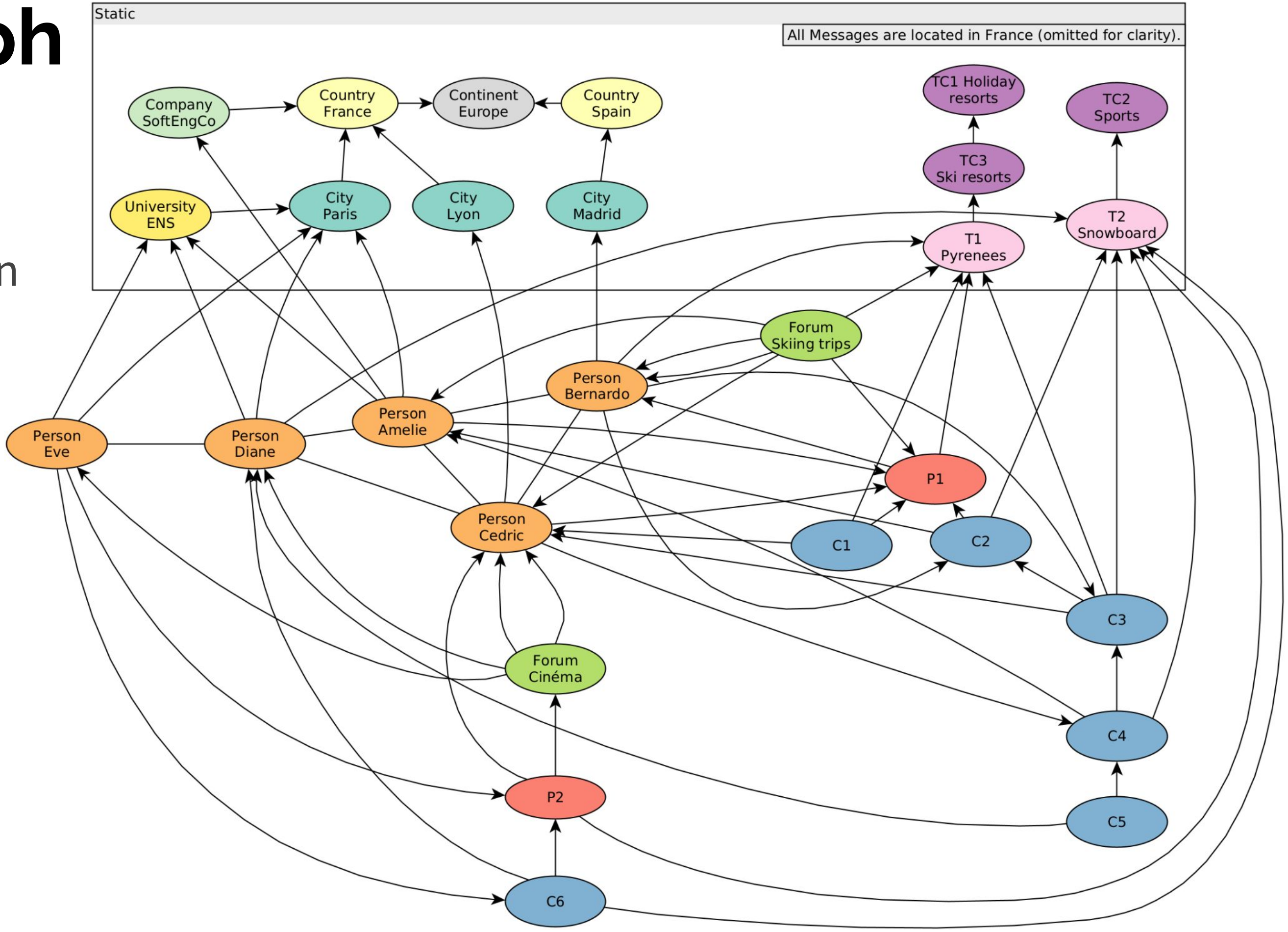
Main entities:

- Person-knows-Person network
- Forums
- Message threads

Correlations:

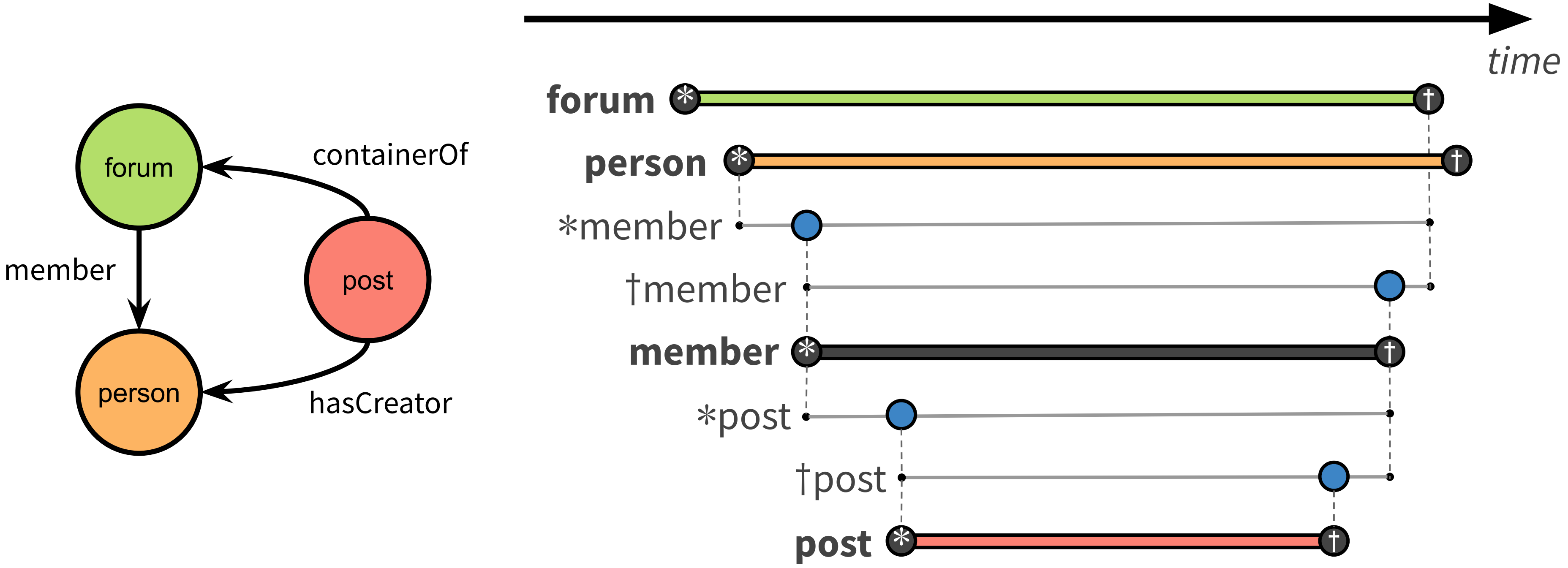
- Structure-level
- Attribute-level

Dynamic graph



Lifespans

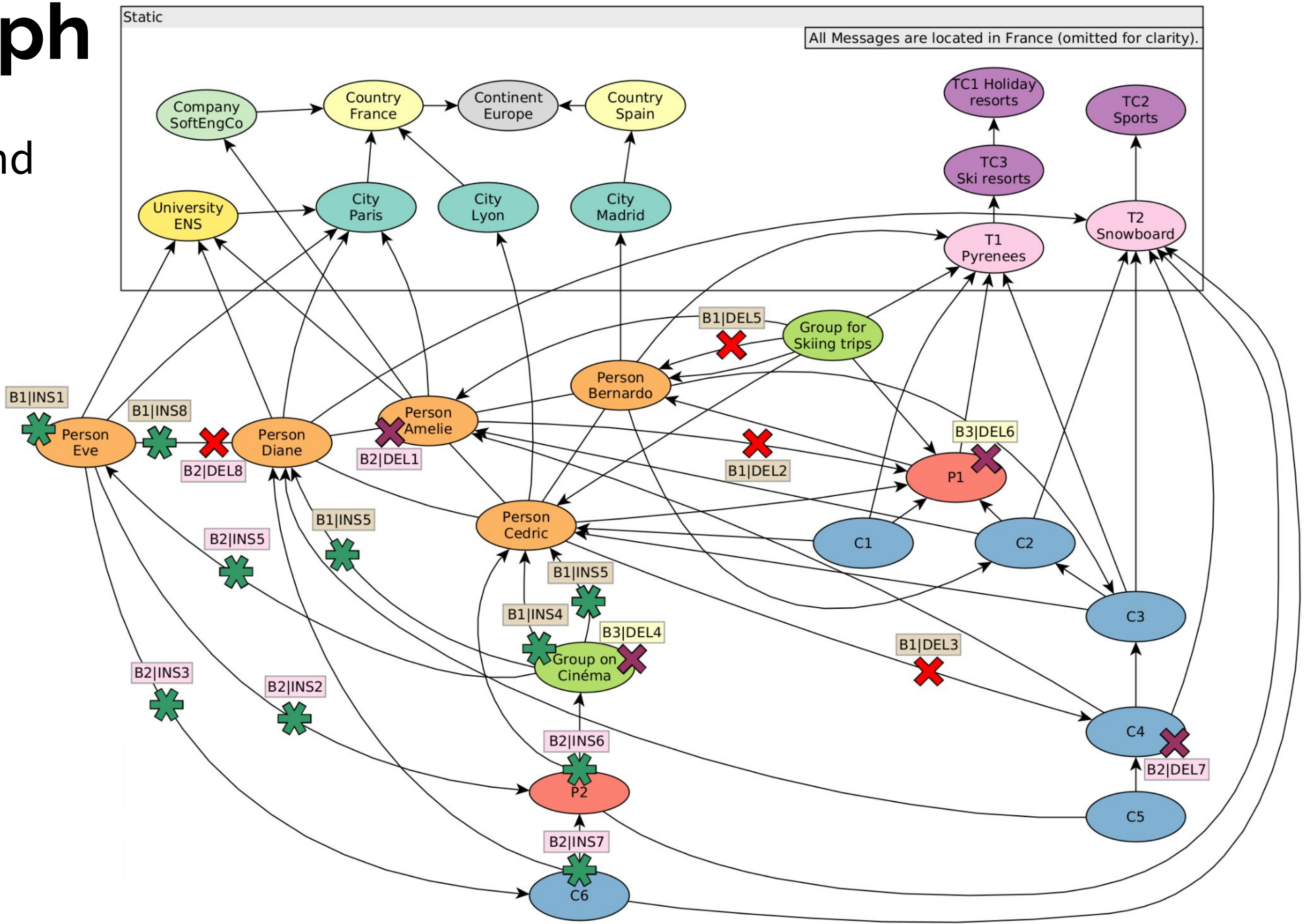
The generator generates the entire temporal with creation dates ∗ and deletion dates †



Dynamic graph

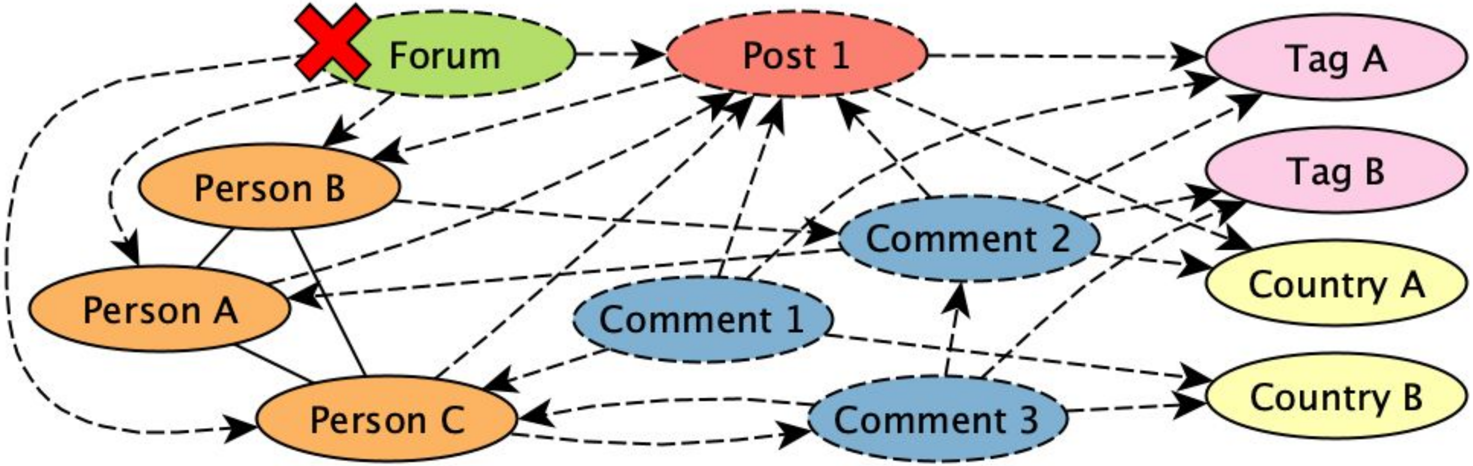
Initial snapshot (97%) and insert/delete batches

📃 [*Supporting dynamic graphs in SNB Datagen*](), GRADES-NDA 2020



Deleting a Forum

Deletes are heavy-hitting operations



Workload

- Workload
- Parameter curation
- Example queries

Choke points

A choke point is a **difficult aspect of query processing** that has a significant impact on the performance of the query.

Examples:

- Join ordering
- Data access locality
- WCOJs
- Path queries

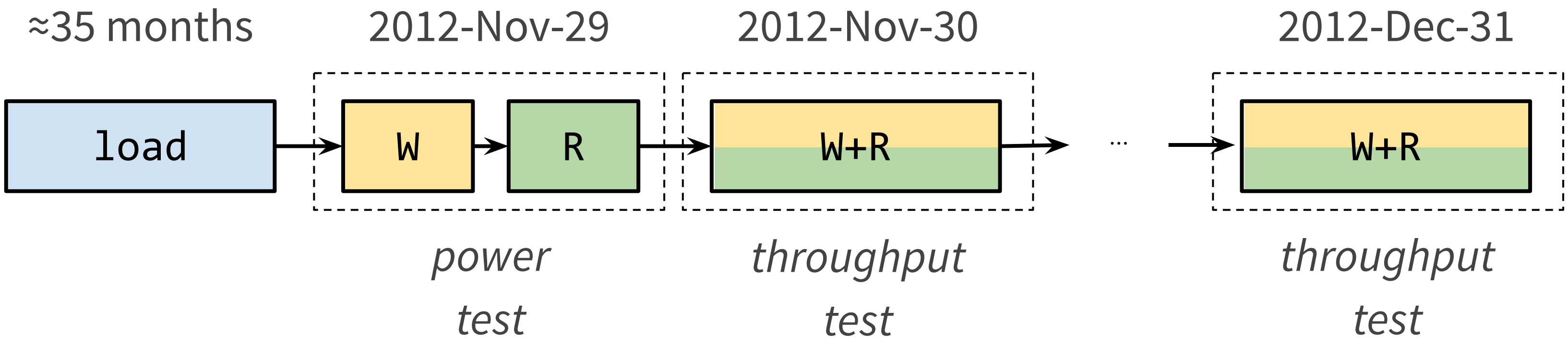
📃 TPCTC’12: experiences of implementing TPC-H on Vectorwise, Virtuoso, and HyPer

Workload

Workload: Ad-hoc graph OLAP queries with daily updates

Batches: 33 days of W/R operations

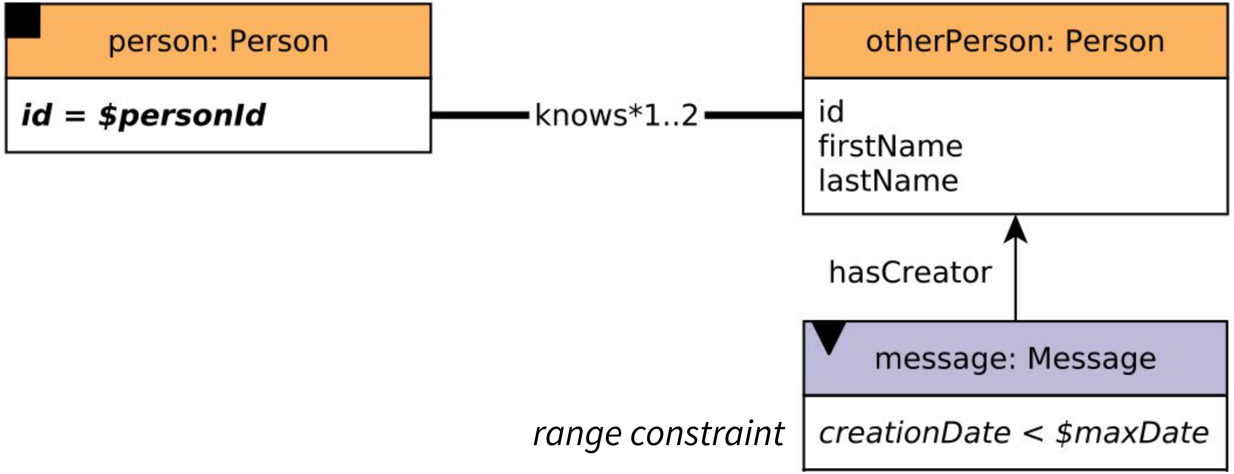
- `W`: apply one day's worth of updates
- `R`: 20 complex read queries with different parameters



Parameter curation

Parameter selection is particularly important for *skewed and correlated data sets*:

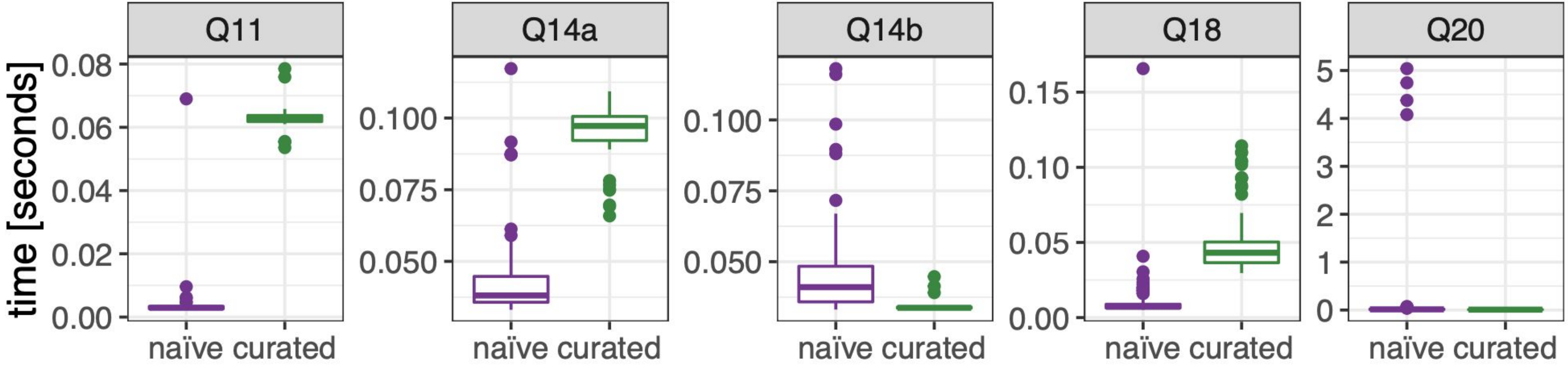
equality constraint



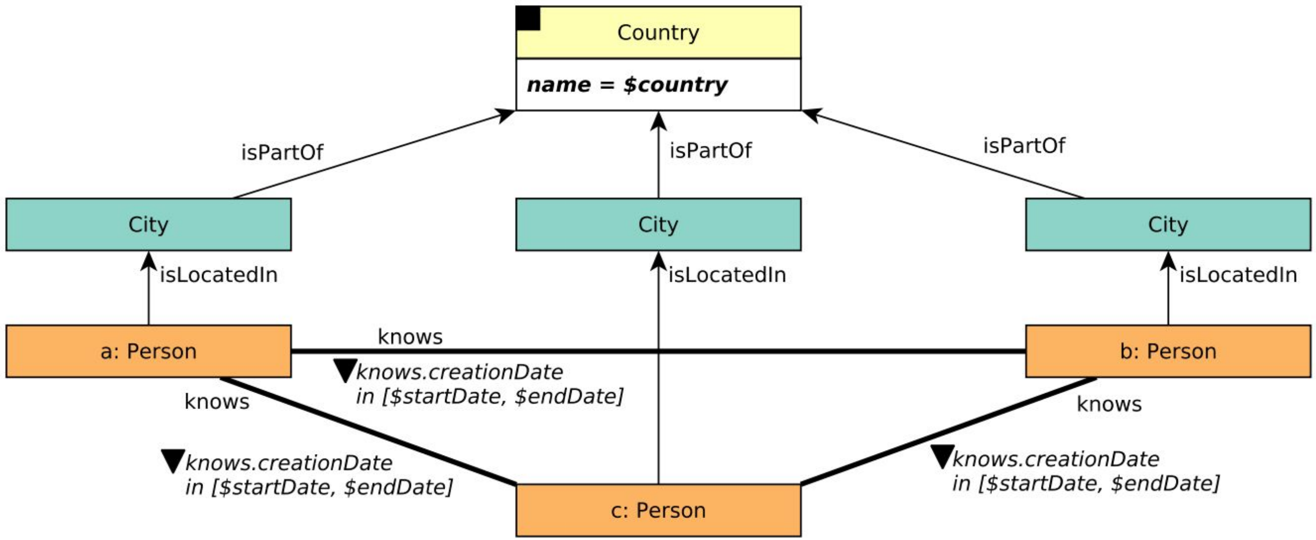
range constraint

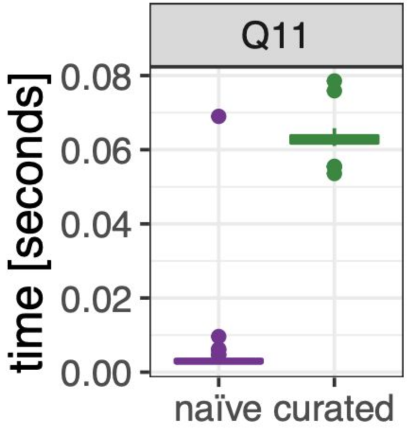
- starting a query from a person with a low degree vs. a high degree
- cost of reachability queries if there is a path vs. no path

Umbra SF10: naïve vs. curated parameters



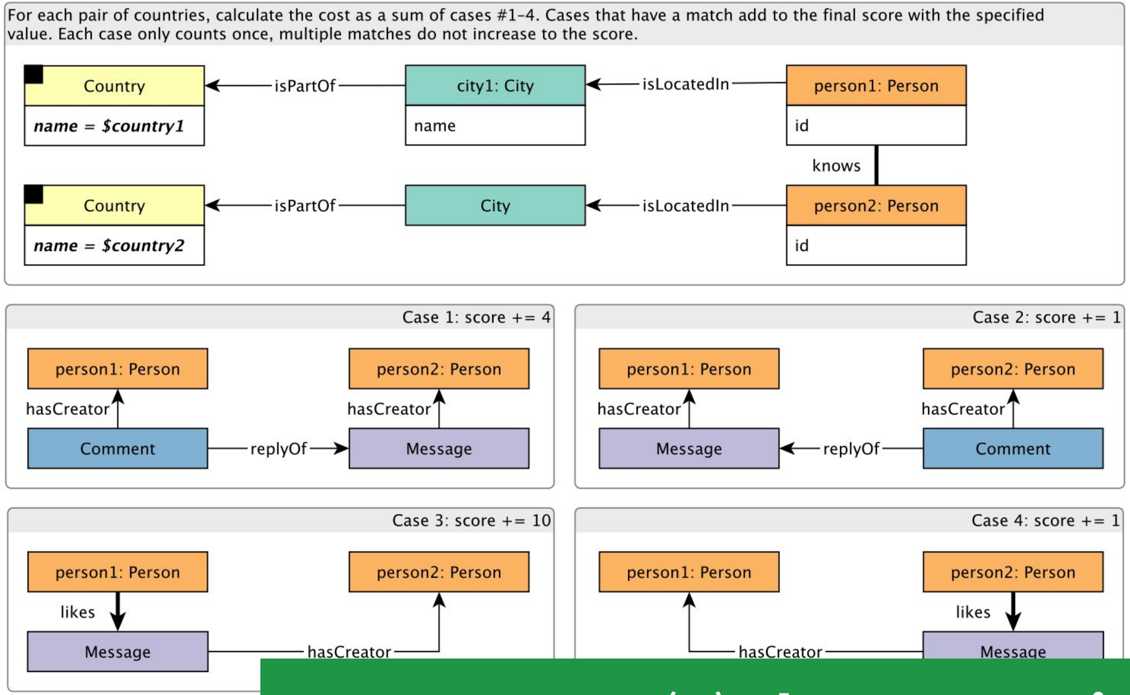
Q11: Triangle query – WCOJs are beneficial

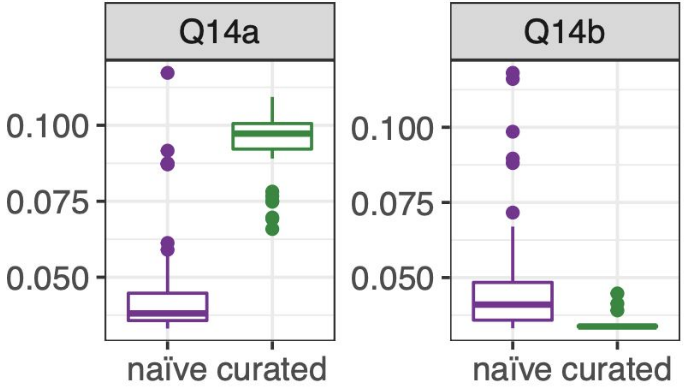




Parameters: Only big countries, similar intervals

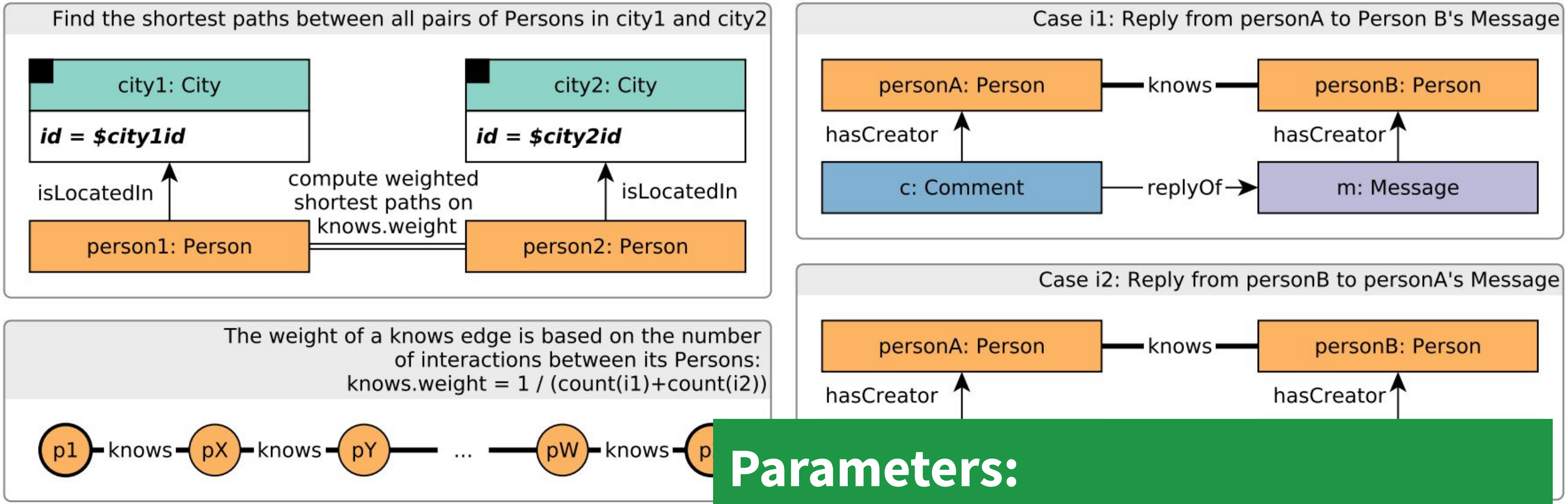
Q14: Correlations – Different runtimes/query plans





Parameters: (A) **close countries** (B) **far-away countries**

Q19: Multi-source weighted shortest path

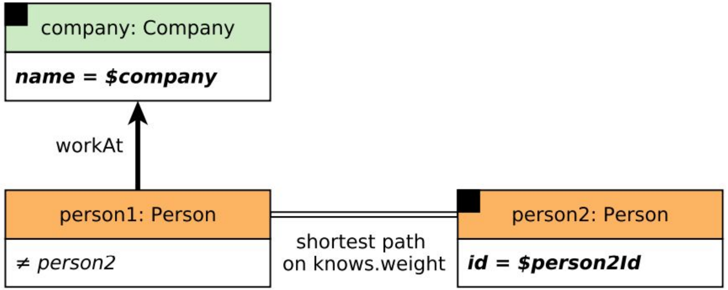


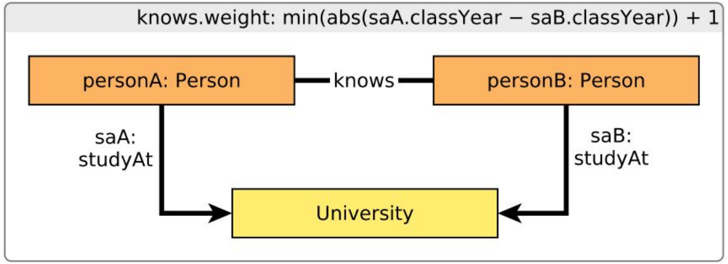
Parameters:

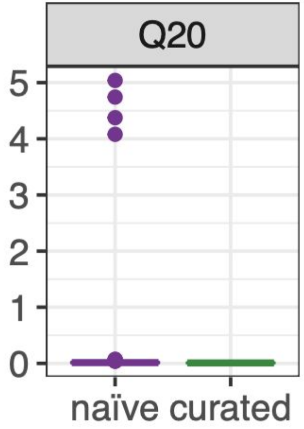
(A) **Cities from the same country**

(B) **Cities from different countries**

Q20: Single-source weighted shortest path







Parameters:

(A) **There is a path between $company employees and $person2**

(B) **There is no path between $company employees and $person2**

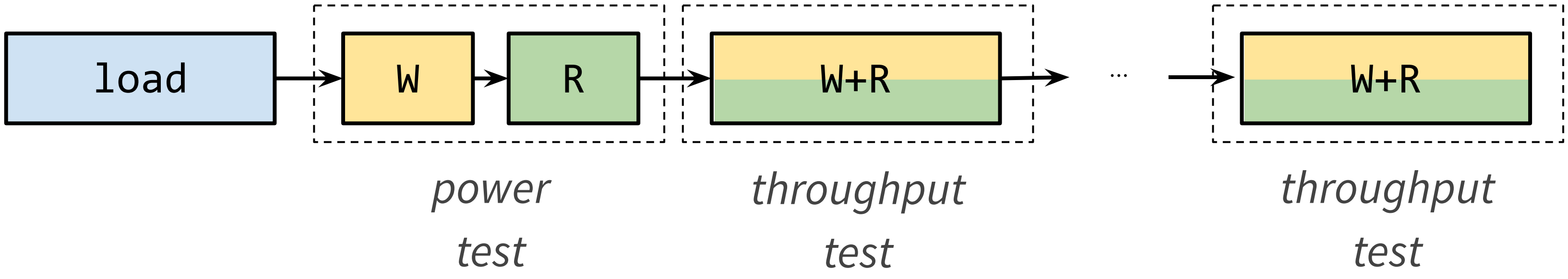
BI implementations

system	data model	language	LOC
neo4j	graph	Cypher	495
UMBRA	relational	SQL	755
TigerGraph	graph	GSQL	832

Execution and scoring

Workload execution

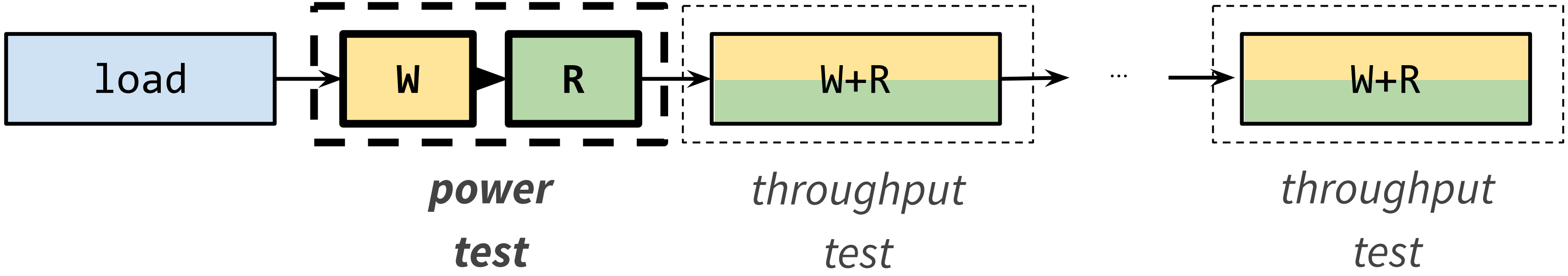
- Power test: sequential query execution
- Throughput tests: concurrent query execution
  - Concurrent RW
  - Disjoint RW



Scoring metrics: Power

Geometric mean ensures all queries are of equal importance

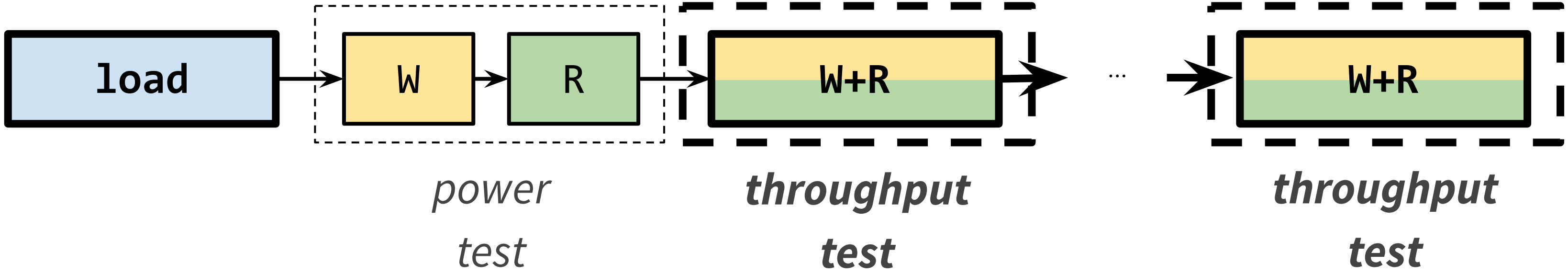
$$power@SF = \frac{3{,}600}{\sqrt[29]{w \cdot q_1 \cdot \ldots \cdot q_{20a} \cdot q_{20b}}} \cdot SF$$



Scoring metrics: Throughput

Run throughput batches for at least 1 hour and extrapolate to one day.

$$throughput@SF = (24 \text{ hours} - t_{load}) \cdot \frac{n_{batches}}{t_{batches}} \cdot SF$$



Scoring metrics: Price-performance

Power and throughput metrics, taking the the total cost of ownership into account, using TPC's pricing.

$$power@SF/\$ = power@SF \cdot \frac{1{,}000}{TCO}$$

$$throughput@SF/\$ = throughput@SF \cdot \frac{1{,}000}{TCO}$$

Scalability

BI workload scales up to SF10k: 10,000 GiB CSV data sets.

- Larger than SF10k results are rare even for TPC-H (~14% in the last decade)

Economics of SF10k generation:

- Data generation: $64
- Parameter generator: <20 minutes on a single machine

Summary

Conclusion

State-of-the-art OLAP benchmark

- Scales to SF10k (10,000 GiB) graphs
- Paper with specification and experiments submitted

Plans:

- Start audits
- Generate SF30k+ data sets
- Backport improvements to SNB Interactive

LDBC

The graph & RDF
benchmark reference

Query design

Choke points and parameters

- Choke point analysis
- Query templates
- Parameter curation

[ENSURE] Scalability

Spark-based data generator for increasing scale factors 1, 3, 10, …

The benchmark *needs to be economical*.

Generating the SF10k data set:

- AWS Elastic MapReduce
- 100 instances with 128GiB RAM
- 1.5 hour runtime

Cost: $74

Comparison of workloads

	Interactive v1.0	Business Intelligence v1.0
focus	OLTP	OLAP
typical query	2-3 hop neighbourhood queries with filtering	multi-hop/path/subgraph queries with filtering & aggregation
refresh operations	inserts	inserts and deletes
target metric	total compression ratio, implying throughput (ops/s)	throughput (ops/day)

Graph data management systems

GDMSs provide a graph-aware UI and support graph processing features.

⚫->⚫ property graph data model

🌀 graph query language

🔎 graph visualization

relational queries

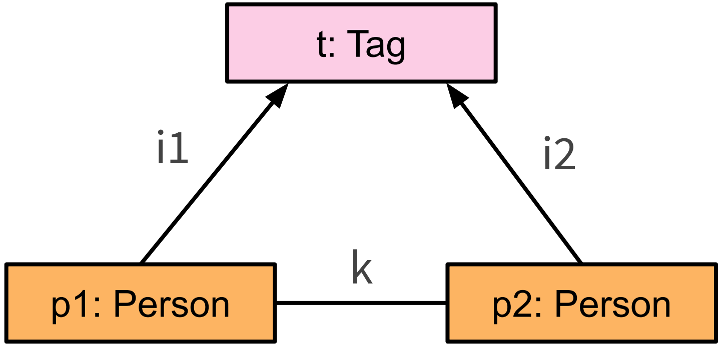
subgraph matching

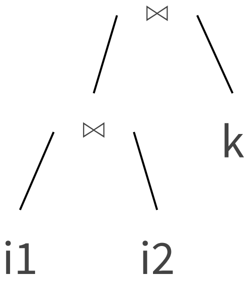
path queries

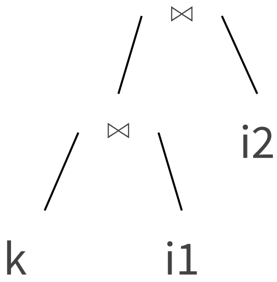
graph algorithms

Subgraph matching

The complexity of **a triangle query with binary joins is provably suboptimal: O(|E|^2)**

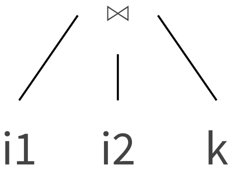






Triggered by many-to-many edges and skewed distributions.

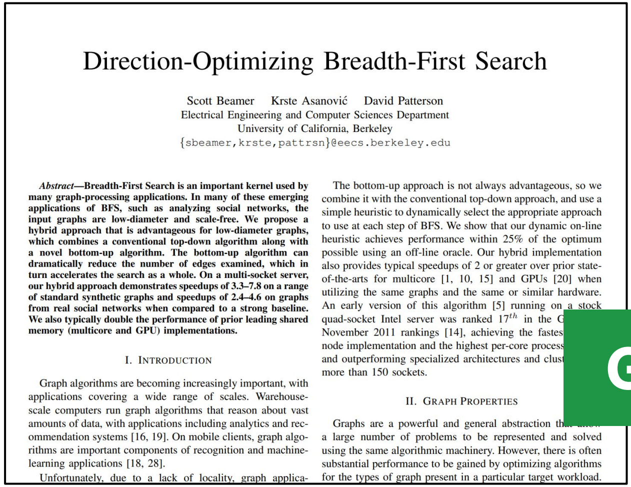
Worst-case optimal **multi-way join algorithms** are needed, which have a complexity of just **O(|E|$^{1.5}$)** for this query.



Path queries

Implementing an efficient BFS/shortest path algorithm is non-trivial:

- direction optimizing BFS (push-pull) — SC 2012
- landmark labelling for distance queries — SIGMOD 2013
- multi-source batched BFS — VLDB 2014


Direction-Optimizing Breadth-First Search

Scott Beamer Krste Asanović David Patterson
Electrical Engineering and Computer Sciences Department
University of California, Berkeley
{sbeamer,krste,pattrsn}@eecs.berkeley.edu

Abstract—Breadth-First Search is an important kernel used by many graph-processing applications. In many of these emerging applications of BFS, such as analyzing social networks, the input graphs are low-diameter and scale-free. We propose a hybrid approach that is advantageous for low-diameter graphs, which combines a conventional top-down algorithm along with a novel bottom-up algorithm. The bottom-up algorithm can dramatically reduce the number of edges examined, which in turn accelerates the search as a whole. On a multi-socket server, our hybrid approach demonstrates speedups of 3.3–7.8 on a range of standard synthetic graphs and speedups of 2.4–4.6 on graphs from real social networks when compared to a strong baseline. We also typically double the performance of prior leading shared memory (multicore and GPU) implementations.

I. Introduction

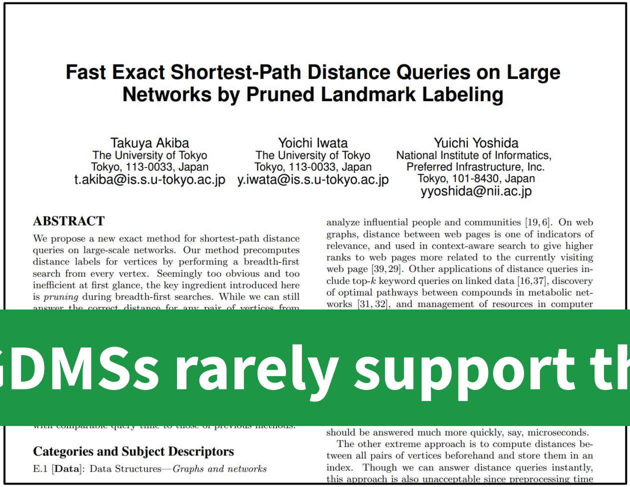
Graph algorithms are becoming increasingly important, with applications covering a wide range of scales. Warehouse-scale computers run graph algorithms that reason about vast amounts of data, with applications including analytics and recommendation systems [16, 19]. On mobile clients, graph algorithms are important components of recognition and machine-learning applications [18, 28].

Unfortunately, due to a lack of locality, graph applica-

The bottom-up approach is not always advantageous, so we combine it with the conventional top-down approach, and use a simple heuristic to dynamically select the appropriate approach to use at each step of BFS. We show that our dynamic on-line heuristic achieves performance within 25% of the optimum possible using an off-line oracle. Our hybrid implementation also provides typical speedups of 2 or greater over prior state-of-the-arts for multicore [1, 10, 15] and GPUs [20] when utilizing the same graphs and the same or similar hardware. An early version of this algorithm [5] running on a stock quad-socket Intel server was ranked 17^{th} in the G… November 2011 rankings [14], achieving the fastes… node implementation and the highest per-core process… and outperforming specialized architectures and clust… more than 150 sockets.

II. Graph Properties

Graphs are a powerful and general abstraction th… a large number of problems to be represented and solved using the same algorithmic machinery. However, there is often substantial performance to be gained by optimizing algorithms for the types of graph present in a particular target workload.


Fast Exact Shortest-Path Distance Queries on Large Networks by Pruned Landmark Labeling

Takuya Akiba, The University of Tokyo, Tokyo, 113-0033, Japan, t.akiba@is.s.u-tokyo.ac.jp
Yoichi Iwata, The University of Tokyo, Tokyo, 113-0033, Japan, y.iwata@is.s.u-tokyo.ac.jp
Yuichi Yoshida, National Institute of Informatics, Preferred Infrastructure, Inc., Tokyo, 101-8430, Japan, yyoshida@nii.ac.jp

ABSTRACT

We propose a new exact method for shortest-path distance queries on large-scale networks. Our method precomputes distance labels for vertices by performing a breadth-first search from every vertex. Seemingly too obvious and too inefficient at first glance, the key ingredient introduced here is *pruning* during breadth-first searches. While we can still …

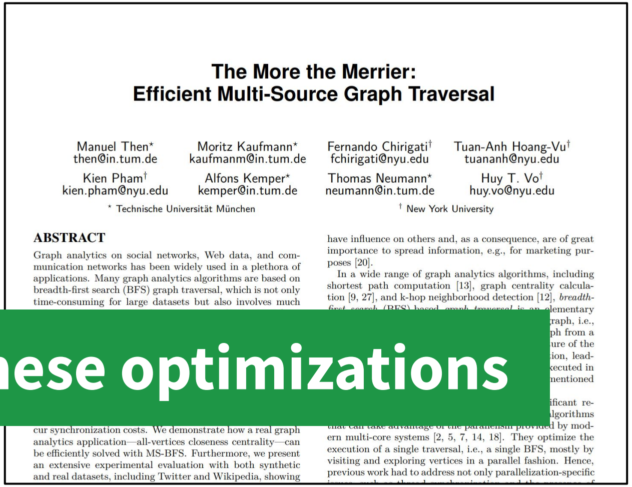
Categories and Subject Descriptors

E.1 [**Data**]: Data Structures—*Graphs and networks*

analyze influential people and communities [19, 6]. On web graphs, distance between web pages is one of indicators of relevance, and used in context-aware search to give higher ranks to web pages more related to the currently visiting web page [39, 29]. Other applications of distance queries include top-k keyword queries on linked data [16, 37], discovery of optimal pathways between compounds in metabolic networks [31, 32], and management of resources in computer …

should be answered much more quickly, say, microseconds.

The other extreme approach is to compute distances between all pairs of vertices beforehand and store them in an index. Though we can answer distance queries instantly, this approach is also unacceptable since preprocessing time …


The More the Merrier: Efficient Multi-Source Graph Traversal

Manuel Then* then@in.tum.de; Moritz Kaufmann* kaufmanm@in.tum.de; Fernando Chirigati† fchirigati@nyu.edu; Tuan-Anh Hoang-Vu† tuananh@nyu.edu
Kien Pham† kien.pham@nyu.edu; Alfons Kemper* kemper@in.tum.de; Thomas Neumann* neumann@in.tum.de; Huy T. Vo† huy.vo@nyu.edu
* Technische Universität München † New York University

ABSTRACT

Graph analytics on social networks, Web data, and communication networks has been widely used in a plethora of applications. Many graph analytics algorithms are based on breadth-first search (BFS) graph traversal, which is not only time-consuming for large datasets but also involves much …

… cur synchronization costs. We demonstrate how a real graph analytics application—all-vertices closeness centrality—can be efficiently solved with MS-BFS. Furthermore, we present an extensive experimental evaluation with both synthetic and real datasets, including Twitter and Wikipedia, showing …

have influence on others and, as a consequence, are of great importance to spread information, e.g., for marketing purposes [20].

In a wide range of graph analytics algorithms, including shortest path computation [13], graph centrality calculation [9, 27], and k-hop neighborhood detection [12], *breadth-first search* (BFS) …

… ern multi-core systems [2, 5, 7, 14, 18]. They optimize the execution of a single traversal, i.e., a single BFS, mostly by visiting and exploring vertices in a parallel fashion. Hence, previous work had to address not only parallelization-specific …

GDMSs rarely support these optimizations

BI implementations

system	data model	language	LOC
Neo4j	graph	Cypher	495
TigerGraph	graph	GSQL	832
Umbra	relational	SQL	755