

Provenance-Based Algorithms for Rich Queries over Graph Databases

EDBT'21

March 25, 2021

Yann Ramusat, Silviu Maniu, Pierre Senellart



Contents

- 1 Preliminaries
- 2 Working Example
- 3 Taxonomy
- 4 Algorithm for absorptive semirings
- 5 Experiments
 - Methodology and datasets
 - Results
- 6 Conclusion

Graph Databases

Graph databases are a common way to manage **graph-like data**.

Notable applications to social network analysis, transportation networks, or the Semantic Web.

Typically queried using **navigational queries**.

We will focus on **Regular Path Queries (RPQs)** (Barceló, 2013)

Provenance annotations

Provenance annotations provide additional information within a graph database to enrich the queries.

These annotations are propagated to query results and can be used to determine:

- how the result has been computed,
- how it reacts to slight changes in the initial database.

Semiring-based provenance

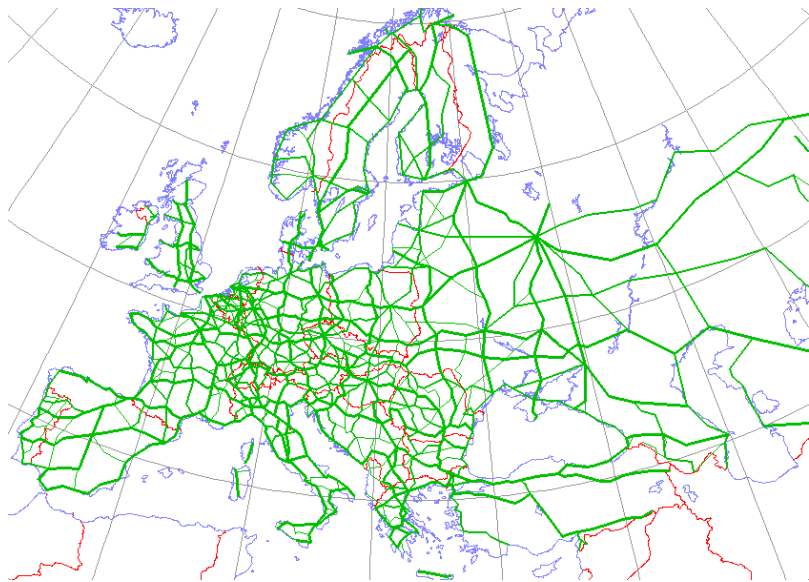
A strong **mathematical foundation** is to choose provenance annotations to be elements of a **semiring** (Green et al., 2007).

Semirings are a well-suited model for **operations** (e.g., choices and sequences) carried along in **computations**.

Ex: the shortest-path problem can be rephrased as solving **equational systems** over semirings.

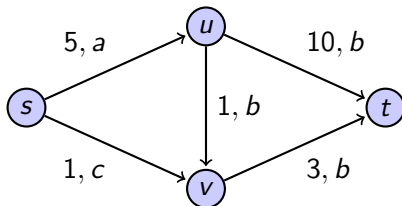
Contents

- 1 Preliminaries
- 2 Working Example**
- 3 Taxonomy
- 4 Algorithm for absorptive semirings
- 5 Experiments
 - Methodology and datasets
 - Results
- 6 Conclusion



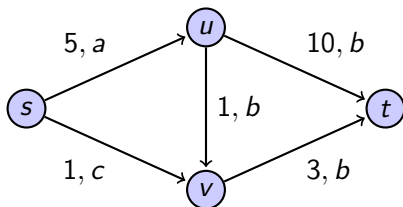
https://en.wikipedia.org/wiki/International_E-road_network

Working example (Tropical semiring)



These integers represent **time to move** between two vertices.
What is the **minimum travel time** between s and t ?

Working example (Tropical semiring)

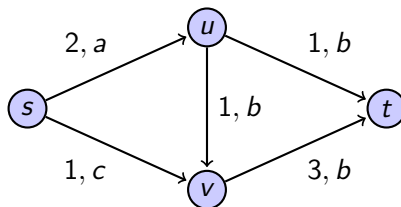


These integers represent **time to move** between two vertices.

What is the **minimum travel time** between s and t ?

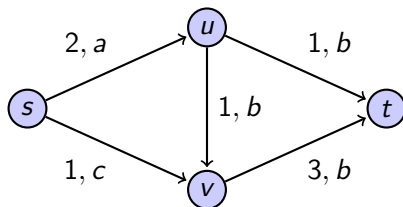
And now, if we **only** want to consider paths going through two b edges?

Working example (Counting semiring)



These integers represent **number of paths** between two vertices.
What is the **total number of paths** between s and t ?

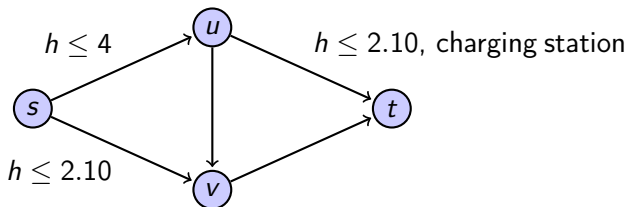
Working example (Counting semiring)



These integers represent **number of paths** between two vertices.

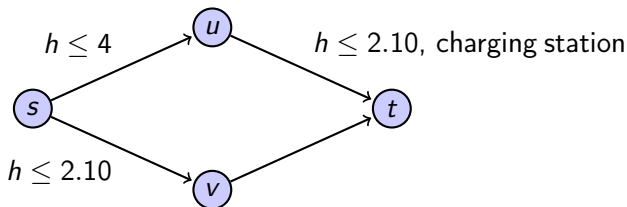
What is the **total number of paths** between s and t ?

And now, if we **only** want to consider paths going through two b edges?

Working example (k -feature semiring)

There exists a path from s to t going through a **charging station**.
And there exists another one permitting **3m high vehicles** to reach t from s .

Working example (k -feature semiring)

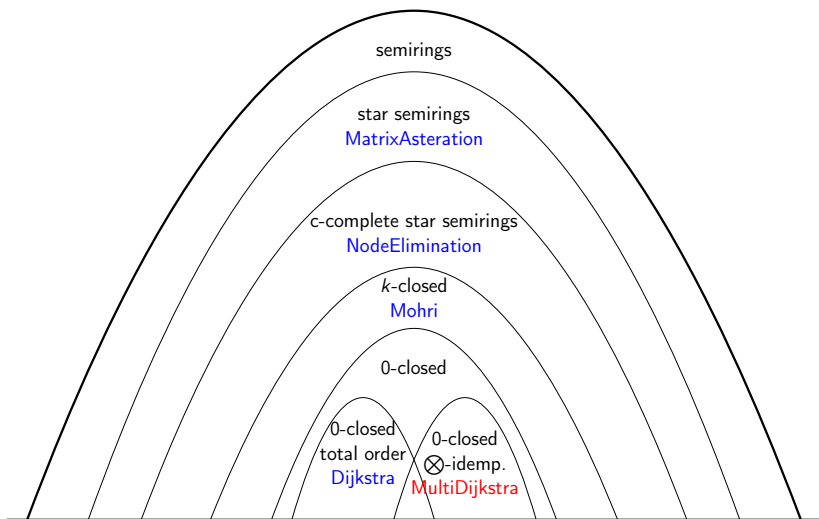


There exists a path from s to t going through a **charging station**.

Contents

- 1 Preliminaries
- 2 Working Example
- 3 Taxonomy**
- 4 Algorithm for absorptive semirings
- 5 Experiments
 - Methodology and datasets
 - Results
- 6 Conclusion

Hierarchy



Complexities

| Name | Semiring property | Time complexity (with provenance) |
|----------------------|--------------------------------|---|
| MatrixAsteration | star | $\mathcal{O}(V T_* + V ^3(T_{\oplus} + T_{\otimes}))$ |
| NodeElimination | c-complete star | $\mathcal{O}(V T_* + V ^3(T_{\oplus} + T_{\otimes}))$ |
| Mohri | k -closed | Exponential |
| MultiDijkstra | 0-closed \otimes -idempotent | $\mathcal{O}(\ell \times (T_{\oplus} V \log V + E (T_{\oplus} + T_{\otimes})))$ |
| Dijkstra | 0-closed total ordered | $\mathcal{O}(T_{\oplus} V \log V + E (T_{\oplus} + T_{\otimes}))$ |

star:

no restriction,

c-complete star:

number of paths,

 k -closed:top- k (distinct) shortest-distances,0-closed \otimes -idempotent k -features,

0-closed total ordered:

access restrictions, shortest-path.

Contents

- 1 Preliminaries
- 2 Working Example
- 3 Taxonomy
- 4 Algorithm for absorptive semirings**
- 5 Experiments
 - Methodology and datasets
 - Results
- 6 Conclusion

0-closed multiplicatively idempotent semirings

Equivalent to **bounded distributive lattices**.

A prominent example of such semiring in the database context is the **PosBool(X)** provenance semiring (Green et al., 2007).

Idea: we consider as a parameter the **width** (ℓ) of the **chain decomposition** of the **join-irreducible** elements of the lattice (Siggers, 2014).

Each element of the lattice can be **uniquely represented** as a **combination** of join-irreducible elements.

Dijkstra can be applied **independently** for each component of the product as they all are **totally ordered**.

MultiDijkstra

Algorithm 1 MultiDijkstra – single-pair

Require: $(G = (V, E, w), s, t)$ a graph database with provenance indication over S , the source s and the target t .

Ensure: Single-pair provenance of the reachability query from s to t .

- 1: **for** each edge $e \in E$ **do**
 - 2: $decompose(w(e))$
 - 3: **end for**
 - 4: **for** each dimension i **do**
 - 5: $d_i \leftarrow Dijkstra(s, t, i)$
 - 6: **end for**
 - 7: **return** $recompose(d_1, \dots, d_n)$
-

Complexity: $\mathcal{O}(\ell \cdot (m + n \log n))$.

Contents

- 1 Preliminaries
- 2 Working Example
- 3 Taxonomy
- 4 Algorithm for absorptive semirings
- 5 Experiments**
 - Methodology and datasets
 - Results
- 6 Conclusion

Methodology and datasets

| type | name | # of vertices | # of edges | tw |
|----------------|-------------|---------------|------------|---------|
| infrastructure | Paris | 4 325 486 | 5 395 531 | 55–521 |
| | Stif | 17 720 | 31 799 | 28–86 |
| | USPowerGrid | 4 941 | 6 594 | 10–18 |
| | Rome99 | 3 353 | 4 831 | 5–50 |
| social | Facebook | 4 039 | 88 234 | 142–237 |
| biology | Yeast | 2 284 | 6 646 | 54–255 |

Only Rome99 has **initial weights** (over TropicalSemiring).

For the others, **weights** are **uniformly generated** between 1 and 3 000.

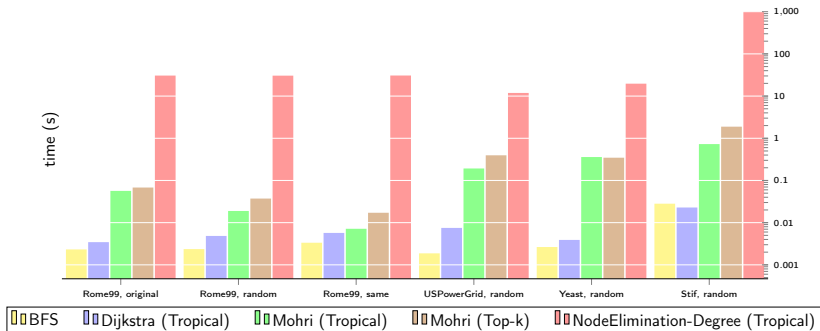
Also use **constant-weight** setting, where all weights equal to 1.

Average over **10 runs** (random choices of **origin** and **destination** vertices).

Datasets from <https://github.com/smaniu/treewidth/>

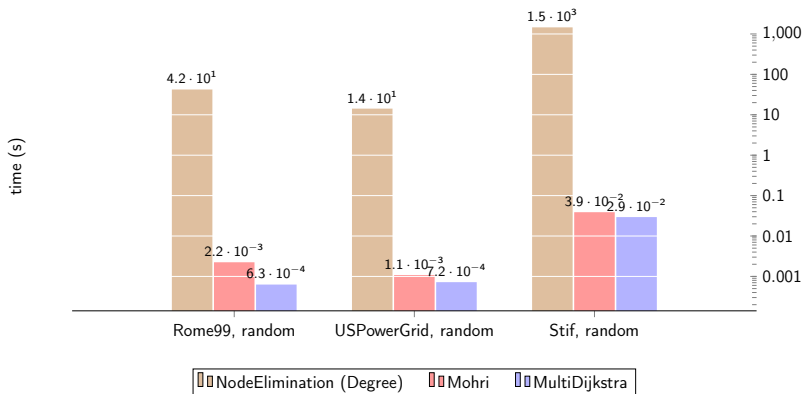
Evaluating shortest distances

Comparison between all of our algorithms for the computation of the **shortest-distance**



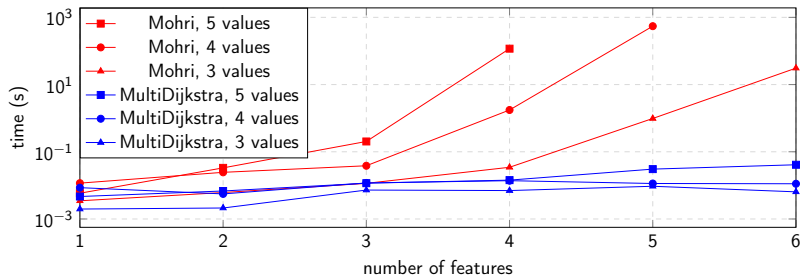
MultiDijkstra

Comparison between NodeElimination, Mohri, and MultiDijkstra
(3-feature semiring)



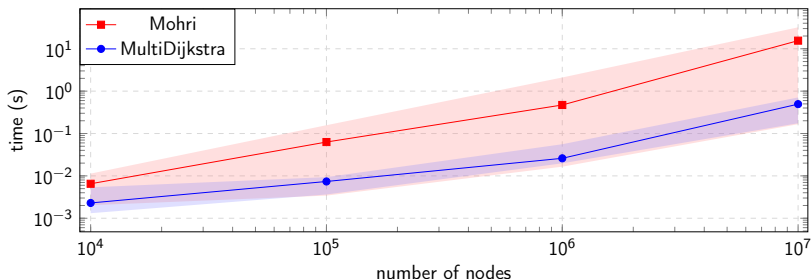
MultiDijkstra

Computation time for Mohri and MultiDijkstra depending on the **number of dimensions** and their **sizes** (Rome99, k -feature semiring)



MultiDijkstra

Average computation time for Mohri and MultiDijkstra over random graphs depending on the number of nodes; shaded areas indicate minimum and maximum computation times observed (3-feature semiring)



Contents

- 1 Preliminaries
- 2 Working Example
- 3 Taxonomy
- 4 Algorithm for absorptive semirings
- 5 Experiments
 - Methodology and datasets
 - Results
- 6 Conclusion

Perspectives

Compare **expressive power** and **efficiency** with Datalog provenance.

Optimization techniques such as Dijkstra's like algorithms can be applied for **hypergraphs** and **context-free grammars** (very close to Datalog).

Striking similarities between important **classes of semiring** in most of these frameworks.

Thanks for your attention.

Bibliography I

- Barceló, P. (2013). Querying graph databases. In *PODS*, pages 175–188, New York. ACM.
- Green, T. J., Karvounarakis, G., and Tannen, V. (2007). Provenance semirings. In *PODS*, pages 31–40, New York. ACM.
- Siggers, M. (2014). On the representation of finite distributive lattices. *arXiv*, 1412.0011 [math].

Complexities

| Name | Semiring property | Time complexity (with provenance) |
|----------------------|--------------------------------|---|
| MatrixAsteration | star | $\mathcal{O}(V T_* + V ^3(T_{\oplus} + T_{\otimes}))$ |
| NodeElimination | c-complete star | $\mathcal{O}(V T_* + V ^3(T_{\oplus} + T_{\otimes}))$ |
| Mohri | k -closed | Exponential |
| MultiDijkstra | 0-closed \otimes -idempotent | $\mathcal{O}(\ell \times (T_{\oplus} V \log V + E (T_{\oplus} + T_{\otimes})))$ |
| Dijkstra | 0-closed total ordered | $\mathcal{O}(T_{\oplus} V \log V + E (T_{\oplus} + T_{\otimes}))$ |

star:

no restriction,

c-complete star:

number of paths,

k -closed:

top- k (distinct) shortest-distances,

0-closed \otimes -idempotent

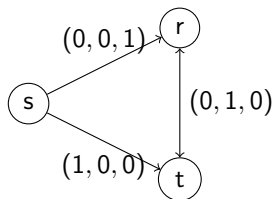
k -features,

0-closed total ordered:

access restrictions, shortest-path.

Counter example to Dijkstra

Using the **3-feature semiring** $(\{0, 1\}^3, \min, \max, (1, 1, 1), (0, 0, 0))$



provenance between s and t is:

$$\min(\max((0, 0, 1), (0, 1, 0)), (1, 0, 0)) = (0, 0, 0)$$

provenance between s and r is:

$$\min(\max((1, 0, 0), (0, 1, 0)), (0, 0, 1)) = (0, 0, 0)$$