Historical Queries on Graphs

Issues and Challenges

Evaggelia Pitoura
Computer Science and Engineering Department
University of Ioannina, Greece
Why?

(Almost) all real-life graphs evolve over time (social networks included)

Both

- **Structure** (nodes, edges)
- **Content** (node and edge properties/content)

How Many Use Facebook?

*Monthly active users at year end*

Source: Company reports
PEW RESEARCH CENTER
Challenge

*Evolving graph*: A sequence of graph snapshots $G^t$ at time instance $t$

Store and process an evolving graph

Both
- Analytical processing and graph mining (community evolution, PageRank, diameter, betweenness, etc)
- *Online* query processing
Query types

Historical queries: queries that refer to the past as opposed to queries that refer to a single (the current) snapshot of the graph

All types of graph queries
- Reachability, Shortest path
- Regular expressions, Graph patterns

- Single snapshot
- Multiple snapshots – interval \([t_1, t_2]\)

semantics [SLP14]: conjunctive (in all), disjunctive (in at least one), at least-\(k\) semantics
  e.g., is \(v\) reachable from \(u\)? degree of \(v\)?
Query types

Queries about the evolution itself

New range of graph queries

E.g.,
- What is the first time that $X$ happened
- The maximum time interval for $X$
- How many times $X$ happened
- What/how much $X$ changed

Historical queries different than queries on dynamic graphs and graph streams
Some issues

What type of queries?

How to store?

How to process queries?

How to index?
Two fundamental approaches [KSP12]

1. Use of *deltas* [KD13]
   - Store (on disk)
   - selected graph snapshots + operational deltas (logs) $\Delta$ (list of operations, e.g., add-edge, delete-edge, etc) $G^1 \Delta = \text{add}(1, 2), \text{delete}(2, 4) \text{ etc}$
   - To create any snapshot $G_t$
   - apply deltas on materialized snapshots

```
G^1
G^2
G^3
...
```
2. Versioning
Annotate the graph (edges, nodes, properties) with validity intervals

A note on partitioning (in memory, parallelism) [HKL+14, LBO+14]:

Two levels of locality:
- Temporal
- Structural
  Query-dependent
Query execution plan

Simple 2-Level Strategy
1. Construct the required snapshots (e.g., apply the deltas, or use a time-index)
2. Use known algorithms

Find-Verify-and-Fix [VLDB11 paper]
Preprocessing
   cluster similar snapshots
   extract two representatives from each cluster ($G_n$ and $G_u$)
1. Apply query to each representative
2. For each graph snapshot $G^t$, verify the solution
3. If not verified, apply query on $G^t$

Partial Reconstruction [KP13]
Egocentric queries:
Restrict snapshots’ reconstruction around a specific node
Index

To avoid online traversal, indexes for specialized graph queries (reachability, shortest-path, patterns)

Example:
For each node $u$ in $G$, a **2 hop-code or label** $(\text{Lin}(u), \text{Lout}(u))$ such that for each pair of nodes $u, v$ in $G$, $v$ is *reachable from* $u$, if and only if,

$$\text{Lin}(u) \cap \text{Lout}(v) \neq \emptyset$$

$L$ **landmarks** $w_1, w_2, ..., w_L$, such that for each pair $u, v$ at least one $w_i$ belongs to their shortest path
For each node $u$, a label $(d(u, w_1), d(u, w_2) ... d(u, w_L))$

$$d(u, v) = \min_i(d(u, w_i) + d(v, w_i))$$
Index

- Single (current) snapshot
- Can we extend them for evolving graphs?
- What is their minimum size?

Shortest-path  [HT14, AIY14 (insert only)]
Our work on historical reachability queries (TimeReach)

- Edge labels are interval sets
- Interval representation using bit vectors
  
  \[[2, 3]\] -> 0110000

- Efficient bit-wise operations for online traversal (time-interval joins)
Our work on historical reachability queries (TimeReach)

- Mapping SCCs across snapshots
- Maintain historical 2HOP for components
Our work on historical reachability queries (TimeReach)

Goal: minimum number of components

Show equivalence with bipartite mapping
Conclusions

There is more than the current snapshot

It is important to look into the type of queries that involve the whole sequence and how to process them
Related work (partial list)


[SLP14] K. Semertzidis, K. Lillis, E. Pitoura: *TimeReach: Indexing for Historical Reachability Queries*, under submission
Thank you! Questions?

Evaggelia Pitoura
Computer Science and Engineering Department
University of Ioannina, Greece
pitoura@cs.uoi.gr