ChronoGraph: Enabling temporal graph traversals for efficient information diffusion analysis over time (Extended abstract)

1st Jaewook Byun
Department of Software
Sejong University
Seoul, Republic of Korea
jwbyun@sejong.ac.kr

2nd Sungpil Woo
Autonomous IoT Research Laboratory
ETRI
Daejeon, Republic of Korea
woosungpil@etri.re.kr

3rd Daeyoung Kim
School of Computing
KAIST
Daejeon, Republic of Korea
kimd@kaist.ac.kr

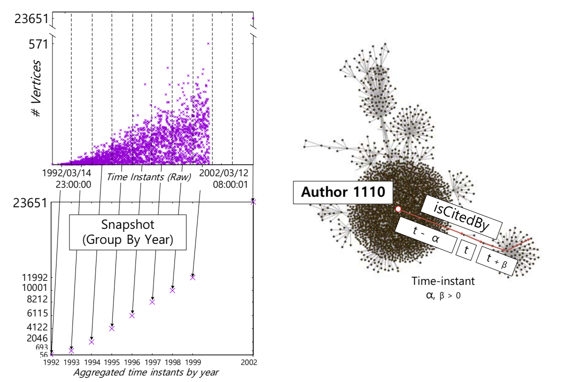
***Abstract*—ChronoGraph is a novel system enabling temporal graph traversals. Compared to snapshot-oriented systems, this traversal-oriented system is suitable for analyzing information diffusion over time without violating a time constraint on temporal paths. The cornerstone of ChronoGraph aims at bridging the chasm between point-based semantics and period-based semantics and the gap between temporal graph traversals and static graph traversals. Therefore, our graph model and traversal language provide the temporal syntax for both semantics, and we present a method converting point-based semantics to period-based ones. Also, ChronoGraph exploits the temporal support and parallelism to handle the temporal degree, which explosively increases compared to static graphs. We demonstrate how three traversal recipes can be implemented on top of our system: temporal breadth-first search (tBFS), temporal depth-first search (tDFS), and temporal single source shortest path (tSSSP). According to our evaluation, our temporal support and parallelism enhance temporal graph traversals in terms of convenience and efficiency. Also, ChronoGraph outperforms existing property graph databases in terms of temporal graph traversals. We prototype ChronoGraph by extending Tinkerpop, a de facto standard for property graphs. Therefore, we expect that our system would be readily accessible to existing property graph users.**

***Index Terms*—ChronoGraph, Temporal Networks, Temporal Graph, Graph Traversal Language, Temporal Aggregation, Parallelism**

I. Introduction

A temporal graph is a graph in which its graph elements (vertices and edges) are only valid at a set of time-instants or time-periods. The temporal graph imposes a graph structure upon temporal data to exploit temporal features inside the data [1].

There are two approaches for temporal network analyses: snapshot-based and traversal-based approach. Existing studies of temporal graph analyses have mainly used a snapshot-based approach due to its accessibility. A snapshot method captures a sequence of static graphs from a temporal graph; thus, existing graph algorithms can be directly applied to each static graph (i.e., snapshot); also, this approach has explored initial temporal graph algorithms (e.g., page rank, shortest path, and connected component). Various graph analytics systems are



(a) Snapshot-based analysis (b) Traversal-based analysis

Fig. 1. An illustration of two approaches to analyze a citation temporal network [2]: (a) The trend of the number of authors in each year (b) Authors who are influenced by the author *1110* over time

proposed for handling a large number of snapshots in real-life graphs. The systems are suitable for analyzing *temporal evolution* analyses. Fig. 1-(a) shows how the number of authors in a citation network evolves over year.

Meanwhile, a traversal-based approach can be a viable alternative way of analyzing temporal networks. In this approach, information from a vertex at a specific time, *vertex event* propagates through temporally valid edges, *edge events*. The direction of information flows is predefined; thus, a temporal path, consisting of footprints with edge events, keeps one of temporal relation [3]. Fig. 1-(b) shows how information of *Author 1110* at $t-\alpha$ spreads through a citation at t to *Author X* where its temporal relation is *isAfter*. Then, the information could spread to other vertices if there are edge events at $t+\beta$ from *Author X*. Algorithms using the traversal-based approach have been proposed: temporal versions of breadth-first search/depth-first search [4] and shortest path techniques [5]. However, to the best of our knowledge, there is no system designed to manage and traverse temporal graphs.

We propose a novel system enabling temporal graph traver-

2375-026X/20/$31.00 ©2020 IEEE
DOI 10.1109/ICDE48307.2020.00232

sals, ChronoGraph. Our main contributions are presented as follows. First, ChronoGraph reconciles point-based semantics and period-based semantics inside temporal data. The coexistence of two semantics should be supported in a graph model, an aggregation method, and a graph traversal language. Second, ChronoGraph exploits the temporal support and parallelism to resolving the time complexity of temporal graph traversals, which is usually far higher than that of static graph traversals. The details can be found in our journal article [6].

II. ChronoGraph

A. Temporal Graph Aggregation

ChronoGraph supports *temporal graph aggregation* to reconcile point-based semantics and period-based semantics inside temporal data because events valid at time-instants and the other events valid at time-periods could coexist. There could be loss of information when we represent period-based events with the point-based representation. In this respect, representing point-based events with the period-based representation can be regarded as a process of restoration. However, this process addresses the issue of how to group discrete time-instant events.

ChronoGraph supports three types of temporal graph aggregation. Firstly, users can use interval between consecutive time-instants, called interval threshold θ. The threshold is the maximum value that weaves two consecutive time-instants t and $t + \alpha$ into one time-period where α is less than equal to θ. Secondly, users can use property values inside an event. According to the given condition, consecutive time-instans would be splitted into one or more time-period(s). Thirdly, users can use a graph structure. From each out-going vertex or in-going vertex, its incident edge events do not have overwrapped time-periods by a given condition.

B. Temporal Syntax on Property Graph

ChronoGraph provides the temporal syntax of de-facto standards for property graphs (i.e., Tinkerpop [7]) in a framework level. Tinkerpop contains a property graph interface, Blueprints and a traversal language, Gremlin. ChronoGraph provides a temporal syntax on Blueprints; thus, the extended interface enables to abstract vertex events and edge events, manipulate the events, and traverse temporal networks via edge events. Also, ChronoGraph provides a temporal syntax on Gremlin; thus, users are able to represent complex temporal graph traversals as a single temporal graph traversal statement.

```
1. Iterable<EdgeEvent> incidentEdgeEvents = g.getVertex("Author 1110")
.getEvent(t-b).getEdgeEvents(Direction.OUT, "isCitedBy", TemporalRelation.isAfter);

2. ve.as("s").scatter().oute("isCitedBy", TemporalRelation.isAfter)
.tFilter(gamma, TemporalRelation.isAfter).gather().tDedup(TemporalRelation.min)
.tStore(gamma).loop("s", exitIfTraverserEmpty).toList();
```

Listing 1: The example of the temporal syntax on a property graph interface and traversal language

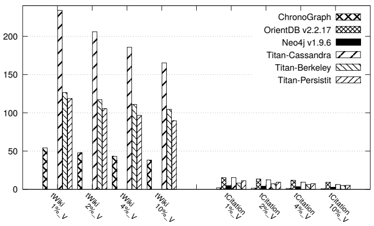


Fig. 2. The comparison of the average tBFS computation time (sec.)

C. Implementation

We prototype ChronoGraph by implementing the temporal syntax on property graphs. Listing 1 shows an example of how users retrieve a vertex event *Author 1110* at $t - \beta$ and traverse to the out-going edge events where their valid time is chronologically larger than $t - \beta$ (1.) and retrieve all the temporally reachable vertices from the source vertex event (2.). Optimizing the methods in an extended interface related to temporal graph traversals could enhance the performance. We manage the temporal indexes for the methods by using additional memory or storage. Also, it is allowed for users to exploit the extended traversal language regardless of parallel processing, path management, and loop management.

III. Conclusion

We propose a novel system for temporal graph traversals, ChronoGraph. Compared to snapshot-based temporal graph systems, our traversal-based system is suitable for analyzing information diffusion over time. In the paper, our focus is the proposal of essential temporal support which tackles the challenges when designing such system dealing with an explosive increase in temporal degree. With the temporal support, ChronoGraph outperforms existing graph databases in terms of temporal graph traversals as shown in Fig. 2.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2018R1C1B5085627, NRF-2019K1A3A1A14012376).

References

[1] V. Kostakos, “Temporal graphs,” *Physica A: Statistical Mechanics and its Applications*, vol. 388, no. 6, pp. 1007–1023, 2009.

[2] J. Leskovec, J. Kleinberg, and C. Faloutsos, “Graph evolution: Densification and shrinking diameters,” *ACM Trans. Knowledge Discovery from Data*, vol. 1, no. 1, pp. 1–40, 2007.

[3] J. F. Allen, “Maintaining knowledge about temporal intervals,” *Communications of the ACM*, vol. 26, no. 11, pp. 832–843, 1983.

[4] S. Huang, J. Cheng, and H. Wu, “Temporal graph traversals: Definitions, algorithms, and applications,” *arXiv preprint arXiv:1401.1919*, 2014.

[5] H. Wu, J. Cheng, S. Huang, Y. Ke, Y. Lu, and Y. Xu, “Path problems in temporal graphs,” *Proceedings of the VLDB Endowment*, vol. 7, no. 9, pp. 721–732, 2014.

[6] J. Byun, S. Woo, and D. Kim, “Chronograph: Enabling temporal graph traversals for efficient information diffusion analysis over time,” *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[7] M. A. Rodriguez, “The gremlin graph traversal machine and language (invited talk),” in *Proceedings of the 15th Symposium on Database Programming Languages*. ACM, 2015, pp. 1–10.