On Mining Temporal Patterns in Dynamic Graphs, and Other Unrelated Problems

Orestis Kostakis[1(✉)] and Aristides Gionis[2]

[1] Microsoft, Redmond, USA
orkostak@microsoft.com
[2] Aalto University, Espoo, Finland
aristides.gionis@aalto.fi

Abstract. Given a social network with dynamic interactions, how can we discover frequent interactions between groups of entities? What are the temporal patterns exhibited by these interactions? Which entities interact frequently with each other before, during, or after others have stopped or started? Such dynamic-network datasets are becoming prevailing, as modern data-gathering capabilities allow to record not only a static view of the network structure, but also detailed activity of the network entities and interactions along the network edges. Analysis of dynamic networks has applications in telecommunication networks, social network analysis, computational biology, and more. We study the problem of mining interactions in dynamic graphs. We assume that these interactions are not instantaneous, but more naturally, each interaction has a duration. We solve the problem of mining dynamic graphs by establishing a novel connection with the problem of mining event-interval sequences, and adapting methods from the latter domain. We apply the proposed methods to a real-world social network and to dynamic graphs from the field of sports. In addition, having established the aforementioned equivalence between the two pattern-mining settings, we proceed to describe how other graph-related problems, such as prediction, learning, and summarization, can be solved by applying out-of-the-box algorithms devised for event-interval sequences. In light of these results, we conjecture that there may be further connections between the two research domains, and the two communities should work closer to share goals and methodology.

1 Introduction

Graphs provide a powerful abstraction for modeling entities and their relations. They are extremely versatile models and are used to represent a wide variety of real-world data. Accordingly, graph mining is currently a very actively-researched topic in Data Mining. Furthermore, since in many applications it is

© Springer International Publishing AG 2018
C. Cherifi et al. (eds.), *Complex Networks & Their Applications VI*,
Studies in Computational Intelligence 689, https://doi.org/10.1007/978-3-319-72150-7_42

possible to collect data that provide detailed information regarding the actions and interactions of the network entities, the topic of mining *dynamic graphs* (or *temporal networks*) has recently emerged [4,9,14,36].

Many ideas have been presented in the literature for mining dynamic or evolving graphs. However, most of the existing work focuses on finding common connected sub-graphs [5], while other methods simply enumerate the observed interactions and return the most frequent [18]. Most importantly, they consider interactions between the entities they represent (the vertices) to be instantaneous. Here we solve a more general problem: given a large dynamic graph where interactions among the graph entities occur at different time instances and have different duration, we aim to find frequent temporal patterns in the interactions between vertices over the lifetime of the graph. Our problem is a generalization of previously studied problems, those of finding frequent sub-graphs or finding frequent interactions; since given the set of frequent patterns we can retain the set of vertices, and edges, correspondingly.

In this paper, we present novel theoretical results that establish a connection between event-interval sequences and *dynamic* graphs. Our results allow to develop methods in order to extract patterns in dynamic graphs using techniques that have been originally designed for event-interval sequences. In particular, we show how to apply sequence-mining techniques in order to discover temporal interaction patterns over a dynamic graph.

The contributions of this work are summarized as follows:

- we demonstrate a direct equivalence between dynamic graphs and event-interval sequences;
- we solve the problem of mining frequent temporal patterns of edge-interactions in dynamic graphs, via mining frequent event-interval sequences;
- as proof of concept, we apply our method and present findings on a real-life social network and dynamic graphs from the field of sports;
- we illustrate the equivalence of additional dynamic graph problems, such as prediction, learning, and summarization, to seemingly unrelated problems on event-interval sequences.

Section 2 contains an overview of the existing literature. The problem that we consider is defined formally in Sect. 3. Based on our theoretical results, which are presented in more detail in Sect. 4, we present proof-of-concept validation results on real-world dynamic graphs in Sect. 5. Finally we describe, in Sect. 6, how additional problems on dynamic graphs may be solved by algorithms that are originally devised for problems on event-interval sequences.

The major aim of this paper is twofold: first, to demonstrate a framework for mining temporal patterns in dynamic graphs. Second, to indicate the strong connection between the fields of Graph Mining and Event-Interval Sequence mining, and to further provoke a deeper discussion among the scientific community concerning the transferability of algorithms and techniques between different subfields of Data Mining, and Computer Science in general.

2 Related Work

The field of dynamic graph mining has received significant attention. Borgwardt et al. [5] devised heuristics to mine a time-series of graphs and extract common sub-graph structures. These sub-graph structures may also exhibit similar behaviour. However, only 'matching' types of behaviours are considered; our work considers 7 relations. GERM [4] was presented for mining graph evolution rules. It is applied to social networks and makes the assumption that the graph can only grow; edges may only be added. It may be extended to handle edge-deletions only if an edge is created and deleted at most once. Polynomial time algorithms for finding periodic subgraphs in a dynamic graph were presented in [19]. TimeCrunch [37] discovers interpretable summaries of dynamic graphs, by it requires providing a predefined set of patterns that should be considered. Similarly, Com2 [3] is limited to directed graphs, and requires defining source- and destination-vertices.

For identifying transition in evolving graphs, Robardet [35] presented an algorithm for identifying five different stages in the lifetime of a large dynamic graph. LEGATO [25] is a framework designed for mining evolving network processes; given a single dynamic network with real-valued edges, the goal is to find a series of sub-graphs that evolve *smoothly* (i.e. consecutive sub-graphs must differ by a bounded number of edges). This work also restricts consecutive subgraphs to share at least one edge.

Dynamic graphs are a natural continuation of static graphs. As a result, many popular problems such as community detection [8], finding maximal cliques [39], detecting and maintaining minimum spanning trees [12], and determining connectivity [13] on dynamic graphs have been solved.

Our work on the equivalence of dynamic graphs and event-interval sequence has been inspired by recent work [15] that demonstrated a strong connection between graphs and event-interval sequences. The latter connection, however, has been shown only for static graphs, and does not directly extend to dynamic graphs.

Most of the work related to event-interval sequences has focused on mining frequent patterns. The majority of existing work on mining frequent event-interval patterns, association rules, and episodes [6,20,27] utilize the *Apriori* algorithm that was designed for mining frequent itemsets [1]. To prune the search space, and thus reduce the complexity of the problem, and yield faster results, subsequent methods have presented tree-based data-structures and more sophisticated techniques for generating candidate patterns [31,32]. For a performance benchmark of some of these methods we refer the reader to a most recent publication [7]. Finally, other efforts have focused on devising distance functions [16,23] and similarity functions [15] for pairs of event-interval sequences.

3 Definitions

3.1 Dynamic Graphs

We consider a set of vertices V. An *interaction* (or an *active edge*) between two vertices $u, v \in V$ during a time interval $[t_s, t_e]$ is denoted by (u, v, t_s, t_e), with $t_s < t_e$. We say that an interaction between u and v is *active* at time-point t if there exists an active edge (u, v, t', t''), with $t' \leq t \leq t''$. An interaction between two vertices may be recurring, that is, two vertices u and v may interact multiple times, freely over time. Hence the *total interaction* between two vertices u and v can be represented as a set $I_{u,v} = \{(u, v, t_s, t_e) \mid u, v \in V\}$. For simplicity, in our model, we consider symmetric interactions, so the total interaction $I_{u,v}$ is considered to be identical to $I_{v,u}$. However, all methods that we describe are directly extendable to handle directionality.

Definition 1. (Dynamic graph) A dynamic graph G is a pair $(V, \mathscr{I})$, where V is a set of vertices, and $\mathscr{I} = \{I_{u,v} \mid u, v \in V\}$ is a set of total interactions between vertices of V.

The *span* (or *lifetime*) of a dynamic graph $G(V, \mathscr{I})$ is the minimal time interval that includes all its interactions; $span(G) = [t_s^*, t_e^*]$, where t_s^* is the minimum starting time-point of any interaction in the graph, and t_e^* is the maximum ending time-point.

As it becomes clear, in our model of the temporal network, we assume that edges (interactions) have a duration. This is a generalization over the case where interactions are instantaneous. For representing instantaneous interactions it is sufficient to set $t_s = t_e$.

A *temporal pattern* of a dynamic graph is a set of interactions (edges) arranged over time in a particular manner. To formally express patterns that contain events with duration, we may use Allen's model [2]. This model defines the temporal relationships among the interactions contained within a pattern. The model defines 13 relations that can be reduced to 7, if we keep one relation from each pair of symmetrical relations; see Fig. 1.

Given a database with dynamic graphs, the *support* of a pattern is the fraction of dynamic graphs in which it occurs. A pattern is *σ-frequent* if its support is greater or equal to a given threshold σ; with $0 < \sigma \leq 1$. We may now formally define the problem addressed in this paper.

Problem 1. Given a ground set of vertices V, a collection of m dynamic graphs $G(V, \mathscr{I}_1), \ldots, G(V, \mathscr{I}_m)$ with non-overlapping spans, and a frequency threshold σ, we aim to find σ-frequent temporal patterns in the interactions of vertices.

The above problem formulation is equivalent to having a single dynamic graph and considering independent (disjoint, non-overlapping) sub-intervals of time; for example days, weeks, months.

In the remainder of this section, we provide the necessary definitions for event-interval sequences.

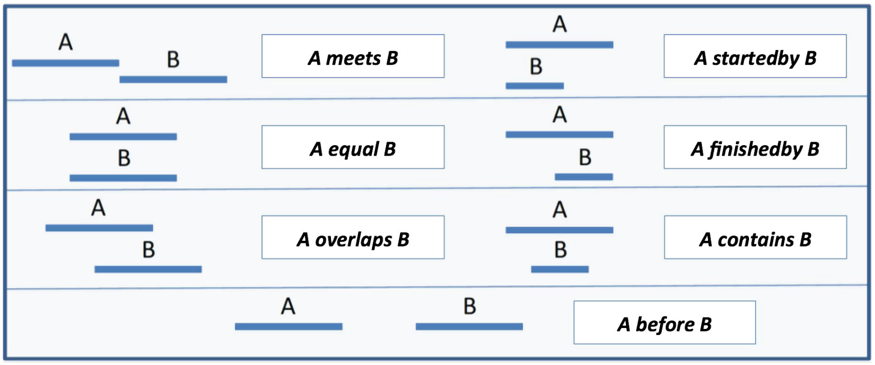

Fig. 1. The seven temporal relations between two intervals, or two interactions of vertices, that are considered in this paper.

3.2 Event-Interval Sequences

Let $\Sigma = \{e_1, \ldots, e_{|\Sigma|}\}$ denote an alphabet of event labels. An event-interval is a triplet $S = (e, t_s, t_e)$, where $e \in \Sigma$ is the event-interval label, and t_s, t_e are the start and end time-points of S. For notational simplicity, we may use $S.t_s$ and $S.t_e$ to denote that start and end time-points of S, respectively.

Definition 2. (e-sequence) An *event-interval sequence*, or *e-sequence* for brevity, $\mathscr{S} = \{S_1, \ldots, S_n\}$ is an ordered set of n event intervals. The temporal order of the event intervals in $\mathscr{S}$ is ascending based on their start time and in the case of ties it is descending based on their end time. If ties still exist, the event intervals are sorted alphabetically.

Figure 2 illustrates an example e-sequence that may be encoded as follows:

$$\mathscr{S} = \{(A, 1, 7), (B, 6, 14), (A, 14, 23), (C, 16, 23), (D, 22, 32)\}.$$

E-sequence $\mathscr{S}$ contains 5 event-intervals, defined over an alphabet of four event-labels.

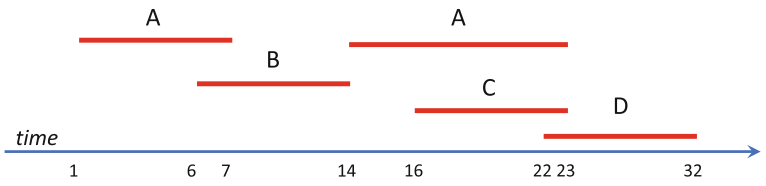

Fig. 2. An example of an e-sequence of five event-intervals defined over an alphabet of four event labels.

Definition 3. (active event) An event $e_i \in \Sigma$ is *active* at time-point t, if there exists an event-interval $S = (e, t_s, t_e)$, such that $S.t_s \leq t$, $S.t_e \geq t$, and $S.e = e_i$.

Apart from the event-duration, the other key attribute of e-sequences is the relations that are formed between pairs of event-intervals. The most popular model for defining those relations is, again, Allen's model [7,15,33].

The reader would notice that relations between intervals are defined by the intervals' starting- and ending-points' relative position. However, the relations do not take into consideration the duration of the intervals. The assumption is that the semantically important information resides in the combinations of interval-relations. On the other hand, this approach is equivalent to allowing, or supporting, time-warping. This problem setting is in accordance with the vast majority of existing work on event-interval sequences [7,15,24,30,32,33].

4 Encoding Dynamic Graphs as Event-Interval Sequences

We demonstrate how dynamic graphs may be encoded as event-interval sequences. Note that the definition of a dynamic graphs is essentially a collection of quadruples, while an e-sequence is defined as a collection of triples. In both cases, the last two coordinates denote the start- and end-time of the interactions, or events. Hence, given a dynamic graph, we may transform it to an e-sequence by assigning to each pair of vertices a unique interval-label; e.g., $(u, v) \rightarrow$ "u_v", and an interaction (u, v, t_s, t_e) becomes an event ("u_v",t_s, t_e). Then, interactions between vertices in a graph translate directly to active intervals in an e-sequence. The combination of all total interactions between vertices in a graph G forms the event-interval sequence S_G.

For the case of graphs with no labels, we can use the same label (a "dummy" label) for each vertex. Then, all intervals would have the same labels. While in this scenario the Apriori-based mining algorithms have to consider fewer choices at each iteration, simpler tasks such as determining the existence of a subsequence within another could require exponential time in the worst case [17].

We have just demonstrated how to traverse from the domain of graphs to the domain of event-interval sequences. The reduction from mining temporal patterns in dynamic graphs to mining event-interval sequences follows directly. We note that the definitions of "closed" and "maximal" patterns [34] carry over between the two applications, too.

By employing frequent e-sequence mining algorithms for discovering patterns in dynamic graphs, we are able to discover patterns that involve all possible vertices, not necessarily restricted to connected components. The vast majority of methods proposed for frequent e-sequence mining apply Apriori-like [1] approaches; if a pattern of size k is not frequent, a super-pattern of size $k + 1$ cannot be frequent. So, when graph-connectivity requirements need to hold, these requirements can be enforced by restricting the algorithm to only consider temporal patterns that correspond to connected components.

Similarly, in many case we can determine the underlying graph-structure of the temporal pattern in a dynamic graph, by modifying the candidate-generation method of any Apriori-based technique for mining event-interval sequences. For example, if we are interested in finding frequent patterns that correspond to star subgraphs of the dynamic network, we need to only expand the candidate at each step with the edges (labels) that contain the central node. For *time-respecting paths* (these are collections of edges that represent a feasible traversal in a dynamic graph from a source vertex s to a destination vertex d) [14], the candidate-generation algorithm would only consider edges containing the latest vertex, but not both vertices that have been already considered. Determining different candidate-generation strategies to acquire different types of underlying graph-structures is left for future work.

We must also point out that while so far we referred to Allen's model, other temporal models are also directly applicable. Moerchen and Fradkin [24] argue that Allen's model is too strict and even minor changes in intervals might distort the results, since the interval relations would change; for example between *overlaps*, *contains* and *finished-by* relations. Hence, they propose the use of *semi-intervals*. In this setting an event-interval can be encoded as the two semi-intervals of its start- and end-point, and the patterns do not need to contain whole intervals. However, we need to point out that semi-intervals, and most simplifications of event-interval sequences to symbolic sequences, are incapable of providing the same richness in representing scenarios; instead, they introduce ambiguities [33].

In the Experiments Section we demonstrate the applicability of both intervals and semi-intervals for mining temporal patterns in dynamic graphs.

5 Experiments

Having demonstrated a connection between the problems of mining temporal interactions in dynamic graphs and mining e-sequences, we provide proof-of-concept evaluation of our approach by applying algorithms for mining e-sequences on dynamic graphs. In particular we use the IEMiner method[1] [33] for finding frequent event-intervals patterns, and the approach of Moerchen et al. [24] for finding frequent semi-interval sequential patterns (SISPs).

Datasets We consider two datasets. The first dataset contains dynamic graphs extracted from professional basketball matches. We collect play-by-play information for 1101 games of the National Basketball Association's (NBA) 2014-15 season [2]. For each game, we analyze the line-ups (group of players in the field) during each possession of the ball. We consider that two players are interacting (there is an active edge between them) if they are simultaneously on the court.

The second dataset is the Reality Mining dataset [11], a social network among 95 mobile-phone users. In this dataset, a pair of users is interacting if one is in the physical proximity of the other. We consider 59 dynamic graphs, each spanning one week, and 24-h time granularity. Users are labeled by their phones' MAC address.

[1] We have used the implementation provided publicly by the author.

[2] We have made the dataset publicly available at: https://goo.gl/uD7a41.

5.1 Results

We apply IEMiner to the 82 games of the Cleveland Cavaliers. The goal is to understand the team's strategy with respect to team-lineups. Figure 3 illustrates a frequent temporal-interval pattern with threshold $\sigma = 0.15$ (the values for σ in this section have been chosen arbitrarily), and the union of those interactions on the graph. In addition, we apply the SISP mining algorithm [24], using the implementation by Vigner et al. [40], to the 55 games of the Dallas Mavericks. The run-time was 28 min. Figure 4 depicts a frequent semi-interval pattern with support threshold $\sigma = 0.55$, and the union of those interactions on the graph.

In both cases, we can confirm that the discovered patterns are meaningful, and in accordance with the teams' rosters, their payroll and common NBA in-game strategies(such as starting with the "best" line-up, and removing all of them before the end of the first half). We also witness the trade-off between well-specified patterns and support when selecting semi-intervals over intervals, as argued by Moerchen et al [24].

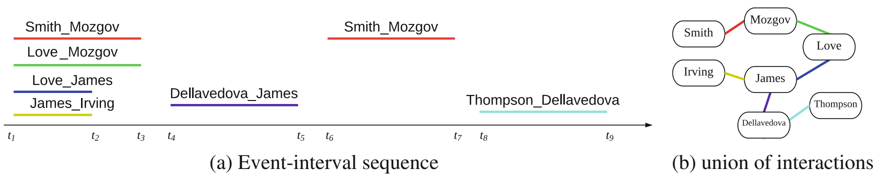

Fig. 3. (a) Frequent interval pattern in the games of Cleveland Cavaliers, support threshold $\sigma = 0.15$, (b) the union of the interactions as a graph.

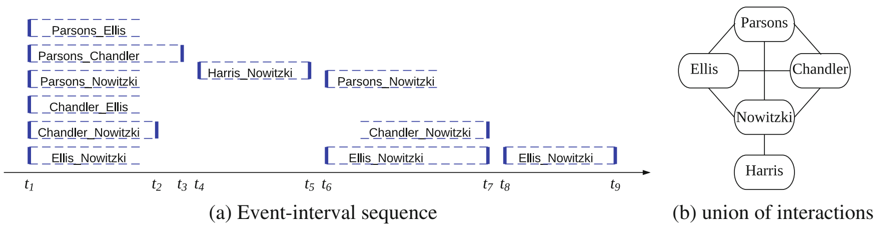

Fig. 4. (a) Frequent semi-interval pattern in the games of Dallas Mavericks, support threshold $\sigma = 0.55$, (b) the union of interactions as a graph.

In addition, we applied IEMiner with support threshold $\sigma = 0.2$ to the Reality Mining dataset. Figure 5a shows an example of a frequent interval pattern; Fig. 5b depicts that pattern as a series of graph "snapshots." The running-time of IEMiner was approximately 4 min.

Since the proposed approach utilizes existing algorithms for mining event-interval sequences, any improvement in terms of running-time and scalability for the latter, directly benefits the task of finding frequent patterns in dynamic graphs.

6 Extension to Other Problems

Next, we describe, when applying our proposed method, how other graph-related problems have equivalent formulations to seemingly unrelated problems on e-sequences.

Pattern Mining. While in this work we experimented with frequent interactions, any other frequent-pattern mining measure can be used. For example, one could replace finding *frequent* patterns with *interesting* patterns [22]; in this scenario the scoring function is different for the generated candidates.

Furthermore, it is also possible to apply the proposed approach to discover association rules between interactions in the graph. These would be in the form of "if there exists a pattern of interactions $\mathscr{A}$ between the set of vertices V, then with confidence c ($0 < c \leq 1$) there exists an additional pattern $\mathscr{B}$". This problem has already been solved for event-interval sequences [32].

Search. Searching within a dynamic graph for a specific temporal pattern, or within a database of dynamic graphs for the existence of similar graphs is exactly the problem of searching for event-intervals sequences under sub-sequence (*Relation-Index* [17]) and full-sequence matching (EBESM [16]), respectively. The demonstrated advantage of the latter methods, in comparison to existing indexing methods, is the ability to handle datasets of very high dimensionality. This is particularly useful with large graphs; multiple vertices hence multiple labels when transforming to event-interval sequences.

Prediction Predicting future interactions in a social network is equivalent to predicting future active event-intervals, and vice-versa. Furthermore, if those interactions are part of a frequent pattern, then existing association rule-mining methods for e-sequences can be applied. In this scenario we would be interested in association rules with the following interval relations: *follow*, *meet*, and in some situations also *overlaps*, *finished-by* and *contains*; because we need to predict interactions into the future. Otherwise to solve the problem of finding/reconstructing missing interactions then we should consider all relations. Conversely, predicting missing event-intervals is equivalent to predicting missing links in a graph. We note that the problems of association rule-mining [32] and event-interval prediction [29] have already been solved.

Learning Unsupervised learning where each data object is a dynamic graph becomes straightforward. We can cluster dynamic graphs based on the similarity of their activity, by utilizing typical distance-based algorithms, such as k-means++ and DBSCAN, together with distance measures devised for e-sequences [16]. Most importantly, our proposed method enables any Data Mining and Machine Learning method that calls as a sub-routine a *distance-* or *similarity-function.*

Summarization Another question, particularly interesting for large graphs, is how to select a strict subset of vertices and interactions of the dynamic graph that retain the most essential information about the graph. In the context of graph mining we are looking for *graph sparsifiers* [21,38], while in the context of e-sequences we are looking for *alphabet-reduction* approaches [26]. In short, any question of the form "can we identify a (small as possible) set of interactions in the dynamic graph, so that we can express/summarize the graph as a linear combination of this set, with minimal information loss?" is identical to summarizing e-sequences by a small set of event-patterns.

Segmentation and Discretization Segmentation is the problem of dividing the span of a dynamic graph into disjoint regions, such that some function is minimized. Usually these functions quantify the homogeneity of the segments. The problem of graph-activity segmentation becomes identical to event-interval sequence-segmentation.

Finally, consider the scenario where the interactions between vertices are continuous functions [10], then many questions regarding discretization of interactions in dynamic graphs are translated to discretization of multi-variate time-series into e-sequences (a process coined *temporal abstraction* [28,30]).

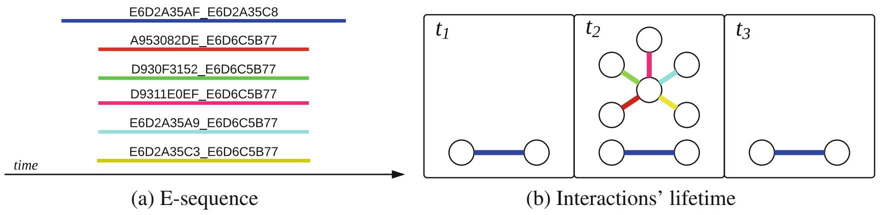

Fig. 5. Frequent interval pattern in Reality Mining, support threshold $\sigma = 0.2$, and the graph interactions over time.

7 Conclusions

We demonstrated a solution for the problem of finding frequent temporal patterns in the interactions within dynamic graphs. In particular, we established how algorithms from a seemingly unrelated problem, those of mining event-interval sequences, may be applied directly to our setting. Apart from the problem of finding frequent temporal patterns, we demonstrated how other problems on dynamic graphs translate directly to problems on event-interval sequences. The insights provided by this paper promote the thesis that the two research communities should work closer to share goals and methodology.

References

1. Agrawal, R., Srikant, R., et al.: Fast algorithms for mining association rules. In: Proceedings 20th VLDB, vol. 1215, pp. 487–499 (1994)
2. Allen, J.F.: Maintaining knowledge about temporal intervals. Commun. ACM **26**(11), 832–843 (1983)
3. Araujo, M., Papadimitriou, S., Günnemann, S., Faloutsos, C., Basu, P., Swami, A., Papalexakis, E.E., Koutra, D.: Com2: fast automatic discovery of temporal (comet) communities. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 271–283. Springer (2014)
4. Berlingerio, M., Bonchi, F., Bringmann, B., Gionis, A.: Mining graph evolution rules. In: ECML PKDD, pp. 115–130 (2009)
5. Borgwardt, K.M., Kriegel, H.P., Wackersreuther, P.: Pattern mining in frequent dynamic subgraphs. In: Proceedings of ICDM, pp. 818–822. IEEE (2006)
6. Chen, X., Petrounias, I.: Mining temporal features in association rules. In: Proceedings of the 3rd European Conference on Principles and Practice of Knowledge Discovery in Databases, pp. 295–300. Springer-Verlag (1999)
7. Chen, Y.C., Weng, J.T.Y., Hui, L.: A novel algorithm for mining closed temporal patterns from interval-based data. Knowl. Inf. Syst. 1–33 (2015)
8. Cordeiro, M., Sarmento, R.P., Gama, J.: Dynamic community detection in evolving networks using locality modularity optimization. Soc. Netw. Anal. Min. **6**(1), 1–20 (2016)
9. Crouch, M., McGregor, A., Stubbs, D.: Dynamic graphs in the sliding-window model. In: ESA, pp. 337–348 (2013)
10. Ding, B., Yu, J.X., Qin, L.: Finding time-dependent shortest paths over large graphs. In: Proceedings of the 11th International Conference on Extending Database Technology: Advances in Database Technology, pp. 205–216. ACM (2008)
11. Eagle, N., Pentland, A.: Reality mining: sensing complex social systems. Pers. Ubiquitous Comput. **10**(4), 255–268 (2006)
12. Henzinger, M., King, V.: Maintaining minimum spanning trees in dynamic graphs. In: Automata, Languages and Programming, pp. 594–604 (1997)
13. Holm, J., De Lichtenberg, K., Thorup, M.: Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. J. ACM (JACM) **48**(4), 723–760 (2001)
14. Holme, P., Saramäki, J.: Temporal networks. Phys. Rep. **519**(3), 97–125 (2012)
15. Kostakis, O., Papapetrou, P.: Finding the longest common sub-pattern in sequences of temporal intervals. Data Min. Knowl. Discov. 1–33 (2015)
16. Kostakis, O., Papapetrou, P.: On searching and indexing sequences of temporal intervals. Data Min. Knowl. Discov. **31**(3), 809–850 (2017)
17. Kostakis, O.K., Gionis, A.G.: Subsequence search in event-interval sequences. In: In Proceedings of ACM SIGIR, pp. 851–854. ACM (2015)
18. Koyutürk, M., Grama, A., Szpankowski, W.: An efficient algorithm for detecting frequent subgraphs in biological networks. Bioinformatics **20**(suppl 1), i200–i207 (2004)
19. Lahiri, M., Berger-Wolf, T.Y.: Mining periodic behavior in dynamic social networks. In: Proceedings of ICDM, pp. 373–382. IEEE (2008)
20. Laxman, S., Sastry, P., Unnikrishnan, K.: Discovering frequent generalized episodes when events persist for different durations. IEEE TKDE **19**(9), 1188–1201 (2007)
21. Mathioudakis, M., Bonchi, F., Castillo, C., Gionis, A., Ukkonen, A.: Sparsification of influence networks. In: Proceedings of ACM SIGKDD, pp. 529–537. ACM (2011)

22. McGarry, K.: A survey of interestingness measures for knowledge discovery. Knowl. Eng. Rev. **20**(01), 39–61 (2005)
23. Meisen, P., Keng, D., Meisen, T., Recchioni, M., Jeschke, S.: Similarity search of bounded tidasets within large time interval databases. In: Computational Science and Computational Intelligence (CSCI), pp. 24–29. IEEE (2015)
24. Moerchen, F., Fradkin, D.: Robust mining of time intervals with semi-interval partial order patterns. In: SDM, pp. 315–326 (2010)
25. Mongiovi, M., Bogdanov, P., Singh, A.K.: Mining evolving network processes. In: Data Mining (ICDM), 2013 IEEE 13th International Conference on, pp. 537–546. IEEE (2013)
26. Monroe, M., Lan, R., Lee, H., Plaisant, C., Shneiderman, B.: Temporal event sequence simplification. IEEE TVCG **19**(12), 2227–2236 (2013)
27. Mooney, C., Roddick, J.F.: Mining relationships between interacting episodes. In: Proceedings of the 4th SIAM International Conference on Data Mining (2004)
28. Mörchen, F., Ultsch, A.: Optimizing time series discretization for knowledge discovery. In: Proceedings of ACM SIGKDD, pp. 660–665. ACM (2005)
29. Moskovitch, R., Choi, H., Hripcsak, G., Tatonetti, N.P.: Prognosis of clinical outcomes with temporal patterns and experiences with one class feature selection. IEEE/ACM TCBB **14**(3), 555–563 (2017)
30. Moskovitch, R., Shahar, Y.: Classification-driven temporal discretization of multivariate time series. Data Min. Knowl. Discov. **29**(4), 871–913 (2015)
31. Moskovitch, R., Shahar, Y.: Fast time intervals mining using the transitivity of temporal relations. Knowl. Inf. Syst. **42**(1), 21–48 (2015)
32. Papapetrou, P., Kollios, G., Sclaroff, S., Gunopulos, D.: Mining frequent arrangements of temporal intervals. KAIS **21**(2), 133–171 (2009)
33. Patel, D., Hsu, W., Lee, M.L.: Mining relationships among interval-based events for classification. In: Proceedings of ACM SIGMOD, pp. 393–404 (2008)
34. Pei, J., Han, J., Mao, R., et al.: Closet: An efficient algorithm for mining frequent closed itemsets. In: ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery **4**, 21–30 (2000)
35. Robardet, C.: Constraint-based pattern mining in dynamic graphs. In: Proceedings of ICDM, pp. 950–955. IEEE (2009)
36. Rozenshtein, P., Tatti, N., Gionis, A.: Discovering dynamic communities in interaction networks. In: ECML PKDD, pp. 678–693. Springer (2014)
37. Shah, N., Koutra, D., Zou, T., Gallagher, B., Faloutsos, C.: Timecrunch: Interpretable dynamic graph summarization. In: Proceedings of ACM SIGKDD, pp. 1055–1064 (2015)
38. Spielman, D.A., Teng, S.H.: Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In: Proceedings of ACM STOC, pp. 81–90. ACM (2004)
39. Stix, V.: Finding all maximal cliques in dynamic graphs. Comput. Optim. Appl. **27**(2), 173–186 (2004)
40. Viger, P.F., Gomariz, A., Gueniche, T., Soltani, A., Wu, C.W., Tseng, V.S.: Spmf: A java open-source pattern mining library. JMLR **15**, 3389–3393 (2014)