Information Systems 51 (2015) 1–26

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

Data structures for temporal graphs based on compact sequence representations ☆

Diego Caro [a,*], M. Andrea Rodríguez [a], Nieves R. Brisaboa [b]

[a] Universidad de Concepción, Chile
[b] Universidade da Coruña, Spain

ARTICLE INFO

Article history:
Received 18 November 2013
Received in revised form
10 November 2014
Accepted 9 February 2015
Recommended by: Xifeng Yan
Available online 17 February 2015

Keywords:
Temporal graphs
Compact data structures
Wavelet tree

ABSTRACT

Temporal graphs represent vertices and binary relations that change along time. In this paper, a temporal graph is conceptualized as the sequences of changes on its edges during its lifetime, also known as temporal adjacency logs. The paper explores the use of compression techniques, and compact and self-indexed data structures, to represent large temporal graphs. More specifically, we present four strategies to represent temporal graphs. The first two strategies, Time-interval Log per Edge (EdgeLog) and the Adjacency Log of Events (EveLog), use compression techniques over the inverted indexes that represent the adjacency logs. Then, we introduce two new strategies to represent temporal graphs using compact and self-indexed data structures. Compact Adjacency Sequence (CAS) represents changes on adjacent vertices as a sequence stored in a Wavelet Tree, and the Compact Events ordered by Time (CET) represents the edges that change in each time instant using Interleaved Wavelet Tree, a new compact and self-indexed data structure specifically designed in this work that is able to represent a sequence of multidimensional symbols (that is, tuples of symbols encoded together). We experimentally evaluate the four strategies and compare them with previous alternatives in the state-of-the-art showing that the four alternatives can represent large temporal graphs making efficient use of space, while keeping good time performance for a wide range of useful queries. We conclude that the use of compression techniques or the use of compact and self-indexed data structures open the possibility for the design of interesting representations of temporal graphs that fit the needs of different application domains.

1. Introduction

Temporal graphs model real networks that exhibit a dynamic behavior where the interactions between elements of the network change over time. For example, consider an online social network where friends are added or removed along time, or a network of mobile communications where connections represent calls between mobile phones. Taking into account the temporal dynamism of graphs allows us to exploit information about temporal correlations and causality, which would be unfeasible through a static (or classical) analysis [1,2]. Classical measures over static

☆ Diego Caro and M. Andrea Rodríguez were funded by Fondef D09I1185. Diego Caro is supported by a CONICYT scholarship for PhD. M. Andrea Rodríguez is funded by Fondecyt 1140428. Nieves Brisaboa is funded by MICINN (PGE and FEDER) Grants TIN2009-14560-C03-02, TIN2010-21246-C02-01 and CDTI CEN-20091048, and by Xunta de Galicia (co-funded with FEDER) ref. 2010/17. We would also like to thank to Diego Seco and José Fuentes for their help in the preliminary discussions of the structures, to Guillermo de Bernardo for his help providing all the *Ks* implementations, and to Claudio Sanhueza from Yahoo! Labs, who helps us with the Flickr dataset.

* Corresponding author. Tel.: +56 41 2204319; fax: +56 41 2221770.
E-mail addresses: diegocaro@udec.cl (D. Caro), andrea@udec.cl (M. Andrea Rodríguez), brisaboa@udc.es (N.R. Brisaboa).

http://dx.doi.org/10.1016/j.is.2015.02.002
0306-4379/

graphs (e.g., centrality, betweenness, and so on) use the assumption that vertices are always connected, which is a naive assumption for real networks. As a consequence, these measures overestimate links availability, which could lead to wrong conclusions about the network.

The main goal of this work is to design strategies to efficiently represent temporal graphs and solve historical queries about the connectivity between their vertices. We consider here data structures that are partial persistent, because they store past data and insert present-time data, but they do not allow past data to be updated [3,4]. Instead of representing temporal graphs by a time-ordered sequence of snapshots, one for each time point, we propose to use a change-based approach, which stores a log of time points when relations between vertices change from inactive to active or vice versa [5]. For precisely, a temporal graph in this work is seen as a set of contacts between vertices, where each contact stores the identifiers of the two vertices and the time interval when these vertices were connected. This has the advantages of storing only what changes and of answering queries about the active direct and reverse neighbors of a vertex. Although there exist few proposals of data structures for temporal graphs based on adjacency lists [6] and in distributed environments [7,8], they have focused on improving time performance for complex algorithms on temporal graphs, overlooking their space cost.

Compact and self-indexed data structures have gained research interest due to their ability to provide indexes to large amount of data using small space. These new structures are known as self-indexes because they store together data and indexes to efficiently answer queries. They have been successfully used for compressing full text indexes [9,10], large graphs [11–15], geographic data [16], binary relations [17], among others. However, despite the increasing interest in compact data structures and in temporal graphs, no much work has been done so far about compact data structures for temporal graphs.

We propose in this work four strategies that represent not only edges of a graph but also the time when changes on the state of these edges occur. This can be seen, therefore, as a sequence of changes on edges, also known as temporal adjacency logs. The four strategies make use of potent techniques of compression, which have been adapted and modified to the problem of temporal graphs. Among them and based on the idea of the Wavelet Tree, we designed and implemented a new Interleaved Wavelet Tree as a structure that stores sequences over a d-dimensional alphabet using $O(nd \log \sigma)$ bits, where n is the length of the sequence and σ is the size of the largest alphabet in any of the d dimensions. The motivation of this structure was to be able to answer direct and reverse queries in temporal graphs with similar time cost. However, this structure can be used in other domains where it is necessary to establish the relationships between different elements of a vocabulary.

The strategies proposed in this work are the following:

- EdgeLog uses a positional inverted index to store a log of the time-varying states of edges, where the lists are compressed with differential codification (d-gaps) and the PForDelta technique [18,19].
- EveLog stores a log of events on edges and also uses an inverted index with differential codification (d-gaps) and the PForDelta technique for the representation of time, and the End-Tagged Dense Code [20] for the representation of vertices (target vertices).
- CAS uses a Wavelet Tree, which is a more sophisticated structure than inverted indexes to represent activation or deactivation of edges, where the sequence in the log is sorted by vertice.
- CET uses the novel structure Interleaved Wavelet Tree and sorts the sequence in the log by time. CET can search for direct and reverse neighbor in a symmetric way and, in addition, and unlike the previous structures, CET is particularly designed to allow queries about what happens at a specific time instant or time interval. Thus, CET can efficiently support not only the same queries than previous structures, but also queries guided by time.

The proposed strategies show a first approach of how the state-of-the-art compression techniques can be used to represent temporal graphs when these graphs are seen as a sequence of changes on the state of edges. This opens a new line of research that is getting more attention due to the need of analyzing networks that evolve in time. The strategies overcome the overload of storing a graph per each time point, knowing as the snapshot or copy strategy [5]. We evaluate the proposals with real and synthetic data in terms of the space and time used to process a set of interesting queries, and with respect to the space used by snapshots represented as a sequence of k^2−trees [15,11], by a snapshot plus a log using the differential k^2−tree [21,22], and by the *FVF-framework* [23].

The organization of this paper is as follows. Section 2 defines temporal graphs, operations on temporal graphs, and previous data structures to represent temporal graphs. Section 3 revises compression techniques and compact self-indexed data structures used in this work. Section 4 presents EdgeLog and EveLog, which are two new strategies that represent temporal adjacency logs based compact inverted indexes. Sections 5 and 6 describe CAS and CET based on compact and self-indexed data structures. Section 7 gives an analytical comparison of structures and the experimental evaluation using synthetic and real datasets. Finally, we present our conclusions and future research directions in Section 8.

2. Background and related work for temporal graphs

In this section we introduce temporal graphs and relevant queries, and also revise previous representations of temporal graphs.

2.1. Temporal graphs

Informally, a temporal graph is a graph where edges can appear or disappear along time [1]. For example, consider how people are subscribed to social networks and how their friendship relations change along time, how devices connect and disconnect in a communication network, or how