ARTICLE IN PRESS

Parallel Computing xxx (2015) xxx–xxx

Contents lists available at ScienceDirect

Parallel Computing

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

Behavioral clusters in dynamic graphs

James P. Fairbanks *, Ramakrishnan Kannan, Haesun Park, David A. Bader

School of Computational Science and Engineering, Georgia Institute of Technology, United States

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Dynamic graph analysis
Streaming
Matrix factorization
Nonnegative Matrix Factorization (NMF)
Behavioral clusters
Low rank approximation

ABSTRACT

This paper contributes a method for combining sparse parallel graph algorithms with dense parallel linear algebra algorithms in order to understand dynamic graphs including the temporal behavior of vertices. Our method is the first to cluster vertices in a dynamic graph based on arbitrary temporal behaviors. In order to successfully implement this method, we develop a feature based pipeline for dynamic graphs and apply Nonnegative Matrix Factorization (NMF) to these features. We demonstrate these steps with a sample of the Twitter mentions graph as well as a CAIDA network traffic graph. We contribute and analyze a parallel NMF algorithm presenting both theoretical and empirical studies of performance. This work can be leveraged by graph/network analysts to understand the temporal behavior cluster structure and segmentation structure of dynamic graphs.

1. Introduction

There are many domains of data analysis that can be modeled with the graph abstraction. In particular we are interested in social networks and internet connection networks. These networks are collections of interactions occurring in complex patterns. Analyzing these patterns is essential to leveraging the information contained in these networks. Because the most important networks are the networks that are in heavy use right now, methods to understand temporal patterns in dynamic networks are important.

The availability of big data has driven an adoption of large scale statistical techniques, both classical and modern. These techniques are not immediately applicable to graph data and this leaves analysts separated from their familiar software tools. In order to connect graph analysis and statistical reasoning, we introduce vertex features which can be calculated efficiently and then analyzed using familiar large scale statistical software tools. This connection is bidirectional because statistical analysis of vertex features informs the computation of additional features. The observed difficulty of writing scalable parallel graph algorithms for scale-free and irregular graphs advises against writing inferential and mathematical code to analyze the graphs directly. In this paper we address this gap by first applying non-inferential graph code to generate vectorial data that is statistically well behaved, then applying a state of the art vectorial technique to this data, which provides insight into the original graph. A representation of this framework is presented in Fig. 1.

In the *massive streaming data analytics model* [11], we view the graph of network events as an unending stream of new edge updates. For each interval of time, we have the static graph, which represents the previous state of the network, and a sequence of edge updates that represent the events since the previous state was recorded. An update can take the form of inserting a new edge, a changing the weight of an existing edge, or a deleting an existing edge. Some networks do not

* Corresponding author.
E-mail addresses: james.fairbanks@gatech.edu (J.P. Fairbanks), rkannan@gatech.edu (R. Kannan), hpark@cc.gatech.edu (H. Park), bader@cc.gatech.edu (D.A. Bader).

http://dx.doi.org/10.1016/j.parco.2015.03.002
0167-8191/

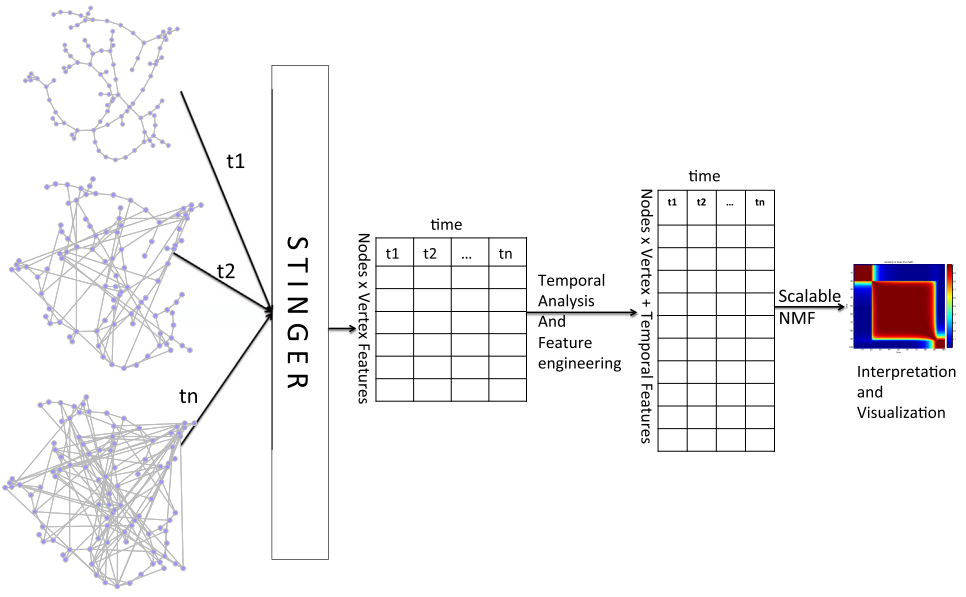

Fig. 1. Our framework combines sparse parallel graph algorithms and dense parallel linear algebra algorithms.

naturally handle deletions, for example Twitter and IP networks where messages are sent and received. In these cases we count the number of messages as the edge weight.

Early work on the theory of streaming algorithms involves summarizing data streams. In a seminal paper by Flajolet and Martin [15], the data is presented in a streaming context and the number of distinct elements must be counted. The algorithms in this field are streaming but the analysis of that data is not necessarily temporal. Feigenbaum et al. [14] have contributed to one model of streaming graph analysis by considering the "semi-streaming model" where graphs are presented "as a stream of edges in adversarial order" and the goal is to compute properties of the graph in one or sub-linearly many passes over the edge stream. This semi-streaming model takes the perspective of a fixed graph with limited access to the data. The work addresses the theoretical issues in computing solutions to "classical graph problems" with necessary approximations due to the constraints on accessing the edges.

With a dynamically changing graph where only those edges occurring in the past can be accessed, there are a new set of temporal queries to answer. Our work contributes to the analysis of modern graph problems that only appear when the edge set is fluctuating over time. We provide insight into applications of temporal data analysis techniques to large data sets that are well represented by the dynamic graph abstraction. Previous approaches to dynamic graph analysis have leveraged traditional, static graph analysis algorithms to compute an initial metric on the graph and then a final metric on the graph after all updates are processed. The underlying assumption is that the time window is large and the network changes substantially so that the entire metric must be recomputed. However, in the massive streaming data analytics model, algorithms react to much smaller changes on smaller time-scales. For example, given a graph with billions of edges, inserting 100,000 new edges might have a small impact on the overall graph, but it might have a large impact on a small subset of the graph. To accommodate this, in our approach an efficient streaming algorithm recomputes metrics on only the affected regions of the graph. For instance, when considering the betweenness centrality of vertices in IP networks each batch of edges represents approximately 36 s of internet traffic. This approach has shown large speed-ups for evaluating clustering coefficients and connected components on scale-free networks [11,13].

In this work we show that the vertex features, as explained above, can be used to generate an understanding of the vertex behavior as well as the behavior of the entire graph as a whole. The nonnegative factorization [1] of these feature matrices provides a clustering of the vertices into groups and a segmentation of the edge stream into phases, which are two important data analysis tasks. These feature matrices are broadly applicable and many applications are beyond the scope of this paper, including tensor factorizations which will provide latent feature based understanding of the three way interaction between the vertices, the different features, and the time-steps.

2. Relevant literature

Previous research has shown that Twitter posts reflect valuable information about the real world. Human events, such as breaking stories, pandemics, and crises, affect worldwide information flow on Twitter. Trending topics and sentiment

[1] Matrix factorization and low rank approximation are used interchangeably for consistency with the literature.

Please cite this article in press as: J.P. Fairbanks et al., Behavioral clusters in dynamic graphs, Parallel Comput. (2015), http://dx.doi.org/10.1016/j.parco.2015.03.002