Expert Systems With Applications 48 (2016) 9–25

Contents lists available at ScienceDirect

Expert Systems With Applications

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

# Enabling graph mining in RDF triplestores using SPARQL for holistic in-situ graph analysis

Sangkeun Lee[a,*], Sreenivas R. Sukumar[a], Seokyong Hong[b], Seung-Hwan Lim[a]

[a] Computational Sciences and Engineering Division, Oak Ridge National Laboratory, TN, USA
[b] Department of Computer Science, North Carolina State University, USA

ARTICLE INFO

*Keywords:*
Graph
Mining
Analysis
RDF
SPARQL
Triplestore
Semantic Web

ABSTRACT

Graph analysis is now considered as a promising technique to discover useful knowledge from data. We posit that there are two dimensions of graph analysis: *OnLine Graph Analytic Processing (OLGAP)* and *Graph Mining (GM)* where each respectively focuses on *subgraph pattern matching* and *automatic knowledge discovery*. As these two dimensions aim to complementarily solve complex problems, *holistic in-situ graph analysis* which covers both OLGAP and GM in a single system is critical for minimizing the burdens of operating multiple graph systems and transferring intermediate result-sets between those systems. Nevertheless, most existing graph analysis systems are only capable of one dimension of graph analysis. In this work, we take an approach to enabling GM capabilities (e.g., PageRank, connected-component analysis, node eccentricity, etc.) in RDF triplestores, which are originally developed to store RDF datasets and provide OLGAP capability. More specifically, to achieve our goal, we implemented six representative graph mining algorithms using SPARQL. The approach allows a wide range of available RDF datasets directly applicable for holistic graph analysis within a system. For validation of our approach, we evaluate performance of our implementations with nine real-world datasets and three different computing environments - a laptop computer, an Amazon EC2 instance, and a shared-memory Cray XMT2 URIKA-GD graph-processing appliance. The experimental results show that our implementation can provide promising and scalable performance for real world graph analysis in all tested environments. The developed software is publicly available in an open-source project that we initiated.

© 2015 Elsevier Ltd. All rights reserved.

## 1. Introduction

The growing interests in discovering relationships across disparate datasets in various domains (e.g., social network, life science, government, healthcare, etc.) have increasingly highlighted graph analysis as a tool for data-driven discovery. *Graph analysis* is a general term describing the process of unveiling useful associations hidden in the data. Major operations in graph analysis fall into two broad categories: (1) online graph query processing such as online data retrieval, aggregation of entities (nodes, edges, paths, and subgraphs) in a graph, and graph pattern matching; and (2) automatic discovery of patterns and prediction of graph properties such as counting triangles (Tsourakakis, 2008), calculating degree distribution, finding eccentricity (Hage & Harary, 1995), finding connected components, and computing PageRank/Personalized PageRank (Page, Brin, Motwani, & Winograd, 1999; Tong, Faloutsos, & Pan, 2006). In this paper, the first category of analysis activities is referred to as *Online Graph Analytic Processing (OLGAP)* and the second category as *Graph Mining (GM)*.

Traditional *On-Line Analytic Processing (OLAP)* (Chaudhuri & Dayal, 1997) performs real-time and online statistical analysis of historical data using a multi-dimensional cube in traditional data warehouse environments. OLGAP can be considered as its counter part. It aims to provide real-time and online data analysis over a graph data model, instead of a multi-dimensional cube. OLGAP encloses graph pattern matching as well as statistical analysis. For example, let us assume that we have a data graph $g$ that was constructed by integrating datasets from two different hospitals A and B. OLGAP answers questions like "Who are the *patients* having the treatment graph pattern $p$ in the healthcare data graph $g$?", where $p$ is usually defined by a subject-matter-expert based on his or her domain knowledge. The expert may perform statistical analysis on the retrieved graph instances, or he can refine the pattern and repeat the process.

On the other hand, instead of utilizing user-defined graph queries, to discover patterns or predict unknown graph properties, GM

* Corresponding author. Tel.: +7188773061.
*E-mail addresses:* lees4@ornl.gov, leesangkeun@gmail.com (S. Lee), sukumarsr@ornl.gov (S.R. Sukumar), shong3@ncsu.edu (S. Hong), lims1@ornl.gov (S.-H. Lim).

http://dx.doi.org/10.1016/j.eswa.2015.11.010
0957-4174/© 2015 Elsevier Ltd. All rights reserved.

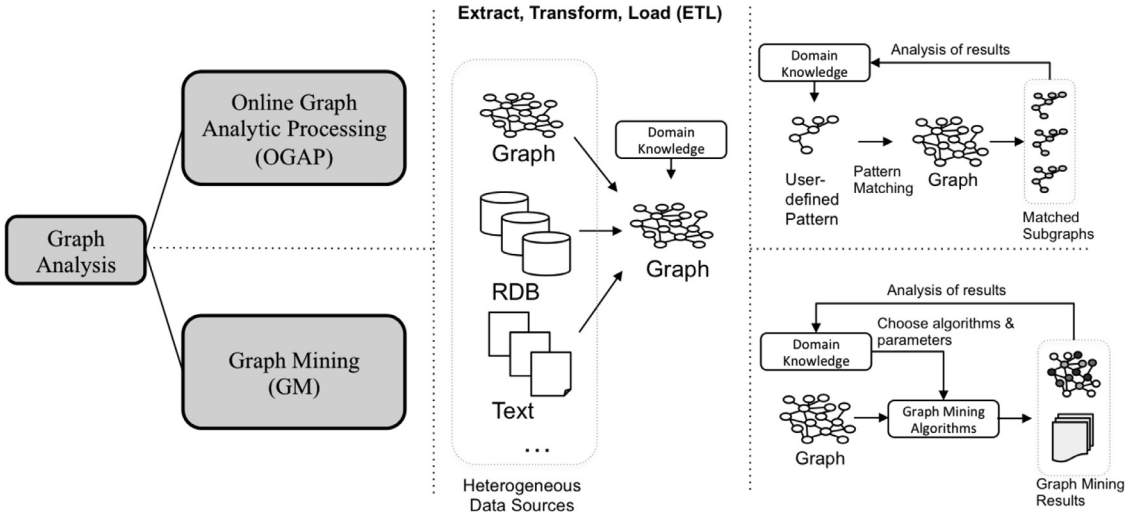


**Fig. 1.** Two aspects of graph analysis - Online Graph Analytic Processing (OLGAP) and Graph Mining (GM).

involves a set of predefined procedures such as *degree distribution, triangle count* (Tsourakakis, 2008), *node eccentricity* (Hage & Harary, 1995), *connected components, PageRank/Personalized PageRank* (Page et al., 1999; Tong et al., 2006), etc. One chooses graph mining algorithms based on the mathematical expression of saliency to answer questions like "Who is likely to be the most influential *person* in the social network?" or "Are there *communities* in the literature citation network?". GM is a sub-category of traditional data mining. Fig. 1 illustrates the relationship between the two major categories of graph analysis.

It is very important to understand that OLGAP and GM complement each other and typically form a cyclic work-flow in graph analysis. For instance, a user may want to compute the importance of nodes in a social network using the *PageRank* algorithm, then they may want to understand the structural friendship pattern among the users with high *PageRank* scores. Thus, supporting both analysis capabilities in a single system, so called the capability of holistic in-situ graph analysis, is critical for data-driven knowledge discovery. The holistic in-situ graph analysis will allow users not to worry about managing multiple systems, dealing with consistency of datasets staged in different systems, and data transformation cost caused by transferring intermediate datasets. However, among a wide range of technologies for graph analysis, it is not easy to find a system that provides both OLGAP and GM capabilities. For instance, *Graph databases* (e.g., Neo4j (Robinson, Webber, & Eifrem, 2013), DEX (Martínez-Bazan et al., 2007), Titan[1], etc.) and RDF *triplestores* (e.g., Jena SDB (Schmidt, Hornung, Küchlin, Lausen, & Pinkel, 2008), Sesame (Broekstra, Kampman, & Van Harmelen, 2002), Virtuoso (Erling & Mikhailov, 2009), etc. ) provide *OLGAP* capability, but they lack built-in GM capabilities. In contrast to graph databases, *graph processing systems* such as Pegasus (Deelman et al., 2005), Giraph (Avery, 2011), and GraphX (Xin, Gonzalez, Franklin, & Stoica, 2013), focus on scaling up graph mining algorithms for large-scale graphs. However, their online pattern matching capability, which is essential for OLGAP, is very limited, as they do not provide high-level graph query processing capability.

In this paper, to address such limitations, we aim to enable *holistic in-situ graph analysis*, which is to perform both OLGAP and GM within a single system. Particularly, we take an approach to enabling GM capabilities in RDF triplestores, which are originally developed to only provide OLGAP capability. More specifically, we implement representative graph mining algorithms using the standard query language of RDF triplestores, called SPARQL.

[1] http://thinkaurelius.github.io/titan/.

Our choice is based on the following reasons. First, there is a number of readily available RDF datasets disseminated by data scientists or organizations. For instance, the *Linked Data Open Community Project* in RDF consists of over 31 billion triples. The open community effort contains a wide range of information about places, people, organisms, diseases, genes, medicines, and vast bibliographic data about books, music, television and movies. Second, thanks to the standardization, various RDF triplestores (e.g., Jena (Schmidt et al., 2008), Sesame (Broekstra et al., 2002), RDFSuite(Alexaki, Christophides, Karvounarakis, Plexousakis, & Tolle, 2001), SPARQLVerse (Liu, Le Calvé, Cretton, & Glassey, 2014), etc.) can take the advantage of our implementations. Note that not only conventional triplestores but also RDF/SPARQL-based graph processing appliance such as Urika-GD (Sukumar & Bond, 2013) can achieve the capability of performing GM. Third, graph datasets that are not in RDF format can be easily transformed in a straightforward way into RDF datasets, as RDF naturally represents a graph. We believe that enabling GM capability in RDF triplestores will bring a huge impact, as it allows a number of data scientists who have their RDF datasets in various RDF triplestore to perform GM without requiring additional data transformation or movement.

Unfortunately, it is not trivial to implement graph mining algorithms using SPARQL. The complexity arises from the fact that most graph-theoretic algorithms have a linear-algebra formulation and assume adjacency-matrices as the default data model. Matrix and array structures are not straightforward to realize using the SPARQL query algebra. One has to redesign algorithms that can handle the triple representation and algorithms have to be simplified for graph operations supported by the SPARQL-query algebra. Also, most graph mining techniques are iterative algorithms and SPARQL does not support iterative query processing.

In our previous workshop paper (Lee, Sukumar, & Lim, 2015b), we introduced initial concepts and challenges of enabling graph mining using SPARQL in triplestores. We presented three iterative graph mining algorithms and evaluated their performance on a standalone machine. As an extension of work, we propose more optimized algorithms along with three additional algorithms in triplestores which are critical to graph analysis. The proposed algorithms are extensively evaluated and analyzed. The main contributions of efforts are summarized as follow:

1. To achieve the capability of *holistic in-situ graph analysis* in RDF triplestores, we present how to implement graph mining algorithms using SPARQL queries. For further development, we

Download English Version:

https://daneshyari.com/en/article/384085

Download Persian Version:

https://daneshyari.com/article/384085

Daneshyari.com