ARTICLE IN PRESS

Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

DPM: A novel distributed large-scale social graph processing framework for link prediction algorithms

Alejandro Corbellini *, Daniela Godoy, Cristian Mateos, Silvia Schiaffino, Alejandro Zunino

ISISTAN-CONICET, UNICEN, Paraje Arroyo Seco - Campus Universitario, CP7000, Tandil, Buenos Aires, Argentina

HIGHLIGHTS

- A novel framework that supports link-prediction algorithms is proposed.
- The programming style is similar to the fork-join style and thus, easy to use.
- Experiments showed that the framework is fast, compared to other two frameworks.
- However, network usage is slightly higher than other frameworks.

ARTICLE INFO

Article history:
Received 20 June 2016
Received in revised form
16 January 2017
Accepted 12 February 2017
Available online xxxx

Keywords:
Distributed graph processing
Recommendation algorithms
Online Social Networks

ABSTRACT

Large-scale graphs have become ubiquitous in social media. Computer-based recommendations in these huge graphs pose challenges in terms of algorithm design and resource usage efficiency when processing recommendations in distributed computing environments. Moreover, recommendation algorithms for graphs, particularly link prediction algorithms, have different requirements depending of the way the underlying graph is traversed. Path-based algorithms usually perform traversals in different directions to build a large ranking of vertices to recommend, whereas random walk-based algorithms build an initial subgraph and perform several iterations on those vertices to compute the final ranking. In this work, we propose a distributed graph processing framework called Distributed Partitioned Merge (DPM), which supports both types of algorithms and we compare its performance and resource usage w.r.t. two relevant frameworks, namely Fork-Join and Pregel. In our experiments, we show that in most tests DPM outperforms both Pregel and Fork-Join in terms of recommendation time, with a minor penalization in network usage in some scenarios.

1. Introduction

The suggestion of friends, contacts or followees in social networks is one of the most prominent problems in today's Online Social Networks (OSNs). This type of recommendation serves multiple purposes, which include reducing users effort in the creation of their own personal networks, improving the quality of user engagement with social sites, favoring information spreading and contributing to the network expansion. In fact, the "Who to Follow" followee recommender service of Twitter is responsible for more than one-eighth of all new connections and it has been one of the major drivers of the company revenue [1].

The problem of suggesting users in an OSN is usually casted to a link prediction problem [2], which tries to infer a non-existent or missing relationship between two persons that is likely to occur in the future. Methods for link prediction use topology-based similarity metrics that can be categorized into path-based, neighbor-based (neighbor-based can be seen as a special case of path-based algorithms of length two) and random walk-based. Several social network recommendation algorithms based on these notions can be found in the literature [1,3,4].

Computing link prediction algorithms on real-world, large-scale social networks poses challenges regarding algorithm scalability considering the inherent huge resource needs (i.e. memory, CPU cores) and performance requirements (e.g. providing fast, real-time recommendations). Most link prediction algorithms,

* Corresponding author.
E-mail addresses: alejandro.corbellini@isistan.unicen.edu.ar (A. Corbellini), daniela.godoy@isistan.unicen.edu.ar (D. Godoy), cristian.mateos@isistan.unicen.edu.ar (C. Mateos), silvia.schiaffino@isistan.unicen.edu.ar (S. Schiaffino), alejandro.zunino@isistan.unicen.edu.ar (A. Zunino).

http://dx.doi.org/10.1016/j.future.2017.02.025
0167-739X/

both commercial ones and those developed in the academia, have been implemented as single-machine, single-threaded applications [1,5]. In consequence, these implementations struggle with scalability issues as the underlying social graph grows, which is commonplace in OSNs populated by a myriad of users.

The natural choice to process large amounts of social data are distributed graph frameworks. In particular, the implementation of link prediction algorithms can be adjusted to different classic distributed processing models, such as MapReduce [6], BSP [7] or, specifically for graphs, Pregel [8]. Such processing models prescribe certain primitives that govern how sub-computations are created and coordinated. However, in terms of graph operations, path-based algorithms have completely different requirements than random walk (RW)-based algorithms. The first ones perform graph traversal operations for a small amount of steps, while the second ones run a successive number of iterations over subgraphs. Hence, counting with the adequate processing framework directly impacts on the performance of the recommendation algorithm.

For graph processing, a generic model such as Fork-Join (FJ) [9] provides a classic divide-and-conquer programming style, in which vertex processing is managed by a parent job that creates and distributes independent tasks and merges their computed results. On the other hand, models like Pregel provide a vertex-centric programming style and distribute the task of merging results among all computing nodes. These frameworks have the disadvantage of being oriented to certain types of algorithms. FJ is usually a good choice for algorithms that execute over a small amount of steps, whereas in the case of iterative algorithms, FJ suffers from the bottleneck produced by the centralized join of results. Pregel, instead, performs well with iterative algorithms but it is not a good fit for algorithms that traverse paths for a small amount of steps. Moreover, algorithm code using the vertex-centric model is usually harder to develop and comprehend due to the message-based communication of results [10].

In this paper, a novel graph processing model called Distributed Partitioned Merge (DPM) is proposed. DPM is a hybrid model since it combines the simplicity of the FJ programming style and the performance and scalability provided by the Pregel framework. In this regard, the main objective of DPM is to quickly compute path-based and RW-based link prediction algorithms, while still providing an easy-to-use programming style as well as a seamlessly integration with a platform specifically designed for providing support for the development of recommender systems in OSNs [11].

In addition, a thorough comparison of FJ, Pregel and DPM for supporting the distributed computation of various link prediction algorithms from the literature was carried out considering the requirements of each type of algorithm (path-based vs. RW-based) in terms of recommendation time and resource consumption. These experiments were performed on a large, real-world, snapshot of the Twitter social network [12] containing 40 million users and 1.4 billion relationships.

The rest of this article is organized as follows. Section 2 discusses related work regarding distributed large-scale graph processing for recommender systems. Section 3 describes FJ, Pregel, and the proposed model. Section 4 reports the experimental setting and results obtained. Finally, conclusions are stated in Section 5.

2. Related work

There are several studies that consider ad-hoc solutions as well as framework-based solutions for distributed graph processing. [13] analyzed the challenges in general-purpose distributed graph processing and considered two ad-hoc implementations over different distributed memory architectures. [14] built an ad-hoc implementation of low-rank approximation of graphs and applied it to link prediction. Unfortunately, without proper background, ad-hoc implementations are usually hard to reuse and maintain.

There are several general-purpose frameworks that have been used to process large-scale graphs in link prediction. Frameworks such as MapReduce [6], Fork-Join [15] or RDD (Spark) [16] have been applied to various graph-related processing problems [17–19]. Other frameworks, like Pregel [8] or GraphLab [20] are specifically designed for graph-based algorithms [21,22]. DPM differs from other frameworks in its ability to support both types of link prediction algorithms.

The comparison of such processing frameworks is difficult due to their design differences. For example, the original MapReduce and Pregel specifications based their failure recovery mechanisms on checkpointing to persistent storage. Thereby, in these frameworks the penalty of using I/O is very high in comparison to RDD-based solutions such as GraphX. Nevertheless, there are some studies that compare processing frameworks, disregarding this unfairness. [23] performed a thorough comparison of the BSP and MapReduce frameworks both in terms of performance and design drivers. Similarly, [24] restricted the performance comparison to Pregel-like framework implementations. The work presented by [25] focused on vertex-centric frameworks (a.k.a. "think like a vertex" frameworks) and makes a distinction from the classic BSP-based frameworks and graph databases. In contrast, the experiments carried out in this work try to establish a common ground of comparison by executing the three selected frameworks (FJ, Pregel and DPM) over the same distributed computing platform [11], sharing the same network communication and graph storage support. The fairness in the comparison is of major importance for determining the right framework for each algorithm.

3. Distributed Partitioned Merge

Developing distributed link prediction algorithms is a difficult task. Thus, the development of these algorithms is often based on distributed processing platforms that provide abstractions that isolate the user from the actual underlying distributed support (both software and hardware). In fact, most distributed platforms expose a programming model or framework that simplifies algorithm development while, at the same time, enforces the correct use of the platform. Even so, the choice of which framework is better for a given algorithm depends on the performance requirements and the ease of use of the programming model. One of the drivers involved in this decision is the type of link prediction algorithm being developed.

For example, path-based link prediction algorithms fit naturally under the divide-and-conquer strategy and, thus, a Fork-Join [15] framework (also known as Split and Merge or Split and Reduce in some frameworks[1]) may be a good fit. The main entity in a distributed FJ algorithm is the FJ job, which represents the computation as a whole. In the fork stage, the FJ job is responsible of creating (or *forking*) parallel tasks to be executed in remote nodes. Once a child task finishes its execution, it sends its results back to the parent job. In this so-called *join* stage, the parent job performs awaits for all the child tasks to finish and then merges their results. The main disadvantage of this framework is that merging sub-results in a single node produces a bottleneck that may negatively impact on algorithms with multiple steps, such as RW-based link prediction algorithms.

[1] For example, GridGain's TaskSplit, http://www.gridgain.com/api/javadoc/org/gridgain/grid/compute/GridComputeTaskSplitAdapter.html.

Please cite this article in press as: A. Corbellini, et al., DPM: A novel distributed large-scale social graph processing framework for link prediction algorithms, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.02.025