Information Sciences 381 (2017) 124–141

Contents lists available at ScienceDirect

Information Sciences

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

Large graph visualizations using a distributed computing platform☆

Alessio Arleo, Walter Didimo*, Giuseppe Liotta, Fabrizio Montecchiani

Dipartimento di Ingegneria, Università degli Studi di Perugia, Italy

ARTICLE INFO

Article history:
Received 12 July 2016
Revised 20 October 2016
Accepted 24 November 2016
Available online 25 November 2016

Keywords:
Large graphs
Network visualization
Force-directed techniques
Distributed algorithms
Giraph
Big data
Cloud computing

ABSTRACT

Big Data analytics is recognized as one of the major issues in our current information society, and raises several challenges and opportunities in many fields, including economy and finance, e-commerce, public health and administration, national security, and scientific research. The use of visualization techniques to make sense of large volumes of information is an essential ingredient, especially for the analysis of complex interrelated data, which are represented as graphs. The growing availability of powerful and inexpensive cloud computing services naturally motivates the study of distributed graph visualization algorithms, able to scale to the size of large graphs. We study the problem of designing a distributed visualization algorithm that must be simple to implement and whose computing infrastructure does not require major hardware or software investments. We design, implement, and experiment a force-directed algorithm in Giraph, a popular open source framework for distributed computing, based on a vertex-centric design paradigm. The algorithm is tested both on real and artificial graphs with up to one million edges. The experiments show the scalability and effectiveness of our technique when compared to a centralized implementation of the same force-directed model. Graphs with about one million edges can be drawn in a few minutes, by spending about 1 USD per drawing with a cloud computing infrastructure of Amazon.

1. Introduction

We live in the so-called *Big Data era*, where huge amounts of data are generated or collected every day through several kinds of devices, sensor networks, information systems, communication channels, social media, and other information science technologies [10]. In this scenario, the use of visualization methodologies and techniques for presenting and analyzing data is rapidly growing and is taking a leading role in conveying information and knowledge to users, especially when data have a complex networked structure, which is typically summarized as a graph [69]. There is a vast amount of research on the design of algorithms and systems for the visualization of large, complex, and dynamic graphs in a variety of application domains (see e.g. [17,18,34,38,45,62,63,67]), including, for example, the analysis of social networks [43,44] and biological networks [19,25]. There are also many user cognitive studies that have been conducted to establish what kinds

☆ Research supported in part by the MIUR project AMANDA "Algorithmics for MAssive and Networked DAta", prot. 2012C4E3KT_001. A preliminary short paper on this research was presented at the 23th Intern. Symposium on Graph Drawing and Network Visualization (GD'15).

* Corresponding author. Fax: +390755853654.
E-mail addresses: alessio.arleo@studenti.unipg.it (A. Arleo), walter.didimo@unipg.it (W. Didimo), giuseppe.liotta@unipg.it (G. Liotta), fabrizio.montecchiani@unipg.it (F. Montecchiani).

http://dx.doi.org/10.1016/j.ins.2016.11.012
0020-0255/

of quality measures have a primary impact on the readability of a graph layout and on the capability of the user to quickly and correctly execute tasks of analysis (see e.g. [2,32,55]).

In the above scenario, classical force-directed algorithms, like *spring embedders*, are by far the most popular graph visualization techniques (see e.g. [39]). One of the key components of this success is the simplicity of their implementation and the effectiveness of the resulting drawings. Spring embedders and their variants make the final user only a few lines of code away from an effective layout of a network. They model the graph as a physical system, where vertices are equally-charged electrical particles that repel each other and edges act like springs that give rise to attractive forces. Computing a drawing corresponds to finding an equilibrium state of the force system by a simple iterative approach (see e.g. [15,21,24]).

The main drawback of spring embedders is that they are relatively expensive in terms of computational resources, which gives rise to scalability problems even for graphs with a few thousands vertices. To overcome this limit, sophisticated variants of force-directed algorithms have been proposed; they include *hierarchical space partitioning, multidimensional scaling, stress-majorization*, and *multi-level* techniques (see e.g. [4,26,28,39] for surveys and experimental works on these approaches). Also, both centralized and parallel multi-level force-directed algorithms that use the power of graphical processing units (GPUs) have been designed and implemented [27,33,60,68]. They scale up to graphs with some million edges, but their development requires a low-level implementation and the necessary infrastructure could be expensive in terms of hardware and maintenance.

Our contribution. The growing availability of powerful and inexpensive cloud computing services [54] naturally motivates the study of distributed graph visualization algorithms, able to scale to the size of large graphs. Indeed, companies are increasingly relying on the use of PaaS (Platform as a Service) infrastructures to process their big data, thus saving the money needed to buy and maintain complex and expensive hardware. We study the problem of designing a simple graph visualization algorithm for a distributed architecture, which can be executed on an inexpensive PaaS infrastructure to compute drawings of graphs with millions of edges. Namely:

- We give a new distributed force-directed algorithm based on the Fruchterman–Reingold model [24], designed according to the "*Think-Like-A-Vertex (TLAV)*" paradigm. TLAV is a vertex-centric approach to design a distributed algorithm from the perspective of a vertex rather than a graph. It improves locality, demonstrates linear scalability, and can be adopted to reinterpret many centralized iterative graph algorithms [50]. Furthermore, it overcomes the limits of distributed paradigms like MapReduce, which are often poor-performing for iterative graph algorithms [40,50].
- We describe an implementation of our algorithm within the *Apache Giraph* framework [14], a popular open-source platform for TLAV distributed graph algorithms. Giraph is used by Facebook to efficiently analyze the huge network of its users and their connections [13]. The code of our implementation is made available over the Web (http://gila.graphdrawing.cloud/), to be easily re-used for further research.
- We present the results of an extensive experimental analysis of our algorithm on a small Amazon cluster of up to 20 computers, each equipped with 4 vCPUs (http://aws.amazon.com/en/elasticmapreduce/). The experiments are performed both on real and artificial graphs, and show the scalability and effectiveness of our technique when compared to a centralized version of the same force-directed model. The experimental data also show the very limited cost of our approach in terms of cloud infrastructure usage. For example, computing a drawing on a set of graphs of our test suite with one million edges requires on average less than 8 min, which corresponds to about 1 USD payed to Amazon.
- Finally, we describe an application of our drawing algorithm to visual cluster detection on large graphs. The algorithm is easily adapted to compute a layout of the input graph using the *LinLog* force model proposed by Noack [53], which is conceived to geometrically emphasize clusters. On this layout, we define and highlight clusters of vertices by using the *K*-means algorithm, and report on the quality of the computed clustering.

Structure of the paper. The remainder of the paper is structured as follows. Section 2 discusses further work related to our research in the context of parallel and distributed graph visualization algorithms. Section 3 provides the necessary background on the force-directed model adopted in our solution and on the TLAV paradigm within Giraph. In Section 4 we describe in details our distributed algorithm, the related design challenges, and its theoretical computational complexity. In Section 5 we present our implementation and experimental analysis. Section 6 shows the application of our algorithm to visual cluster detection on large graphs. Finally, in Section 7 we discuss future research directions for our work.

2. Parallel and distributed graph visualization algorithms

So far, the design of parallel and distributed graph visualization algorithms has received limited attention.

- Calamoneri et al. [8] described a simple and efficient parallel algorithm for a restricted class of graphs; namely, given an n-vertex cubic graph G, the algorithm computes an orthogonal drawing of G in $O(\log n)$ time using n processors on an EREW PRAM.
- Mueller et al. [52] and Chae et al. [9] proposed force-directed algorithms that use multiple large displays. Vertices are evenly distributed on the different displays, each associated with its own processor, which is responsible for computing the positions of its vertices; scalability experiments are limited to graphs with some thousand vertices.