Chapter 5
Large Scale Graph Mining With MapReduce:
Diameter Estimation and Eccentricity Plots of Massive Graphs with Mining Applications

Charalampos E. Tsourakakis
Carnegie Mellon University, USA

ABSTRACT

In recent years, a considerable amount of research has focused on the study of graph structures arising from technological, biological and sociological systems. Graphs are the tool of choice in modeling such systems since they are typically described as sets of pairwise interactions. Important examples of such datasets are the Internet, the Web, social networks, and large-scale information networks which reach the planetary scale, e.g., Facebook and LinkedIn. The necessity to process large datasets, including graphs, has led to a major shift towards distributed computing and parallel applications, especially in the recent years. MapReduce was developed by Google, one of the largest users of multiple processor computing in the world, for facilitating the development of scalable and fault tolerant applications. MapReduce has become the de facto standard for processing large scale datasets both in industry and academia.

In this chapter, the authors present state of the art work on large scale graph mining using MapReduce. They survey research work on an important graph mining problem, estimating the diameter of a graph and the eccentricities/radii of its vertices. Thanks to the algorithm they present in the following, the authors are able to mine graphs with billions of edges, and thus extract surprising patterns. The source code is publicly available at the URL http://www.cs.cmu.edu/~pegasus/.

DOI: 10.4018/978-1-61350-513-7.ch005

INTRODUCTION

The total digital output is expected to exceed 1.2 ZetaBytes in 2010 (Blake, 2010). The New York Stock Exchange generates about one terabyte of new trade data per day and Facebook hosts approximately 10 billion photos, taking up one PetaByte of storage (White, 2009). It has become apparent that as the amount of data generated increases at this unprecedented rate, scalability of algorithms is crucial. In recent years, MapReduce (Dean et al., 2008) and Hadoop (Hadoop Wiki, 2010), its open source implementation, have become the *de facto* standard for analyzing large datasets. Despite its limitations, the MapReduce framework stands out for making the programmer's life who uses MapReduce to develop applications easy. Specifically, from the programmer's perspective, MapReduce is just a library imported at the beginning of the program, like any other common library. MapReduce takes care of the parallelization and all its details including distributing the data over the cluster and fault tolerance. In the next Section we provide more details on MapReduce and Hadoop. According to (Hadoop Users, 2010) over 70 major companies over the world use Hadoop. Furthermore, innovative commercial ideas like Amazon's Elastic Compute Cloud (EC2) where users can upload large data sets and rent processor time in a large Hadoop cluster have proved successful. Besides companies, MapReduce and Hadoop have become also the *de facto* standard for research. Several universities including Carnegie Mellon University, Cornell and Berkeley are using Hadoop clusters for research purposes. Projects include text processing, analysis of large astronomical datasets and graph mining. Currently, (Pegasus CMU, 2010) provides an open source Hadoop-based library for performing important graph mining operations.

In this Chapter, we survey state-of-the-art work related to estimating the diameter in massive graphs using MapReduce. The interested reader is urged to study the original publications (Kang et al., 2010a), (Kang et al., 2010b) which we survey in this Chapter for the full details of the algorithms described in the following. The outline of this Chapter is as follows: in Section 2 we provide a brief description of MapReduce and Hadoop, an open source package which includes a freely available implementation of MapReduce. Furthermore, we present the necessary background for the proposed method HADI (HAdoop DIameter). In Section 3 we present HADI and in Section 4 we show certain applications of our method. In Section 5 we provide future research directions. Finally, in Section 6 we conclude. For the interested reader, we provide at the end of the Chapter additional reading material.

BACKGROUND

In this section we provide the necessary background: the MapReduce framework, basic graph theoretic definitions and the Flajolet-Martin method for counting distinct elements in a multiset.

MapReduce

While the PRAM model (Jaja, 1992) and the bulk-synchronous parallel model (BSP) (Valiant, 1990) are powerful models, MapReduce has largely "taken over" both industry and academia (Hadoop Users, 2010). In few words, this success is due to two reasons: first, MapReduce is a simple and powerful programming model which makes the programmer's life easy. Secondly, MapReduce is publicly available via its open source version Hadoop. MapReduce was introduced in (Dean et al, 2008) by Google, one of the largest users of multiple processor computing in the world, for facilitating the development of scalable and fault tolerant applications. In the MapReduce paradigm, a parallel computation is defined on a set of values and consists of a series of *map*, *shuffle* and *reduce* steps. Let $(x_1,.., x_n)$ be the set of values, *m* denote the mapping function which