International Journal of Approximate Reasoning 55 (2014) 896–907

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

www.elsevier.com/locate/ijar

A comparison of parallel large-scale knowledge acquisition using rough set theory on different MapReduce runtime systems ☆

Junbo Zhang [a,b], Jian-Syuan Wong [b], Tianrui Li [a,*], Yi Pan [b]

[a] School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, China
[b] Department of Computer Science, Georgia State University, Atlanta, GA 30303, USA

ARTICLE INFO

Article history:
Received 19 November 2012
Received in revised form 5 May 2013
Accepted 20 August 2013
Available online 17 September 2013

Keywords:
Rough sets
Knowledge acquisition
MapReduce
Large-scale

ABSTRACT

Nowadays, with the volume of data growing at an unprecedented rate, large-scale data mining and knowledge discovery have become a new challenge. Rough set theory for knowledge acquisition has been successfully applied in data mining. The recently introduced MapReduce technique has received much attention from both scientific community and industry for its applicability in big data analysis. To mine knowledge from big data, we present parallel large-scale rough set based methods for knowledge acquisition using MapReduce in this paper. We implemented them on several representative MapReduce runtime systems: Hadoop, Phoenix and Twister. Performance comparisons on these runtime systems are reported in this paper. The experimental results show that (1) The computational time is mostly minimum on Twister while employing the same cores; (2) Hadoop has the best speedup for larger data sets; (3) Phoenix has the best speedup for smaller data sets. The excellent speedups also demonstrate that the proposed parallel methods can effectively process very large data on different runtime systems. Pitfalls and advantages of these runtime systems are also illustrated through our experiments, which are helpful for users to decide which runtime system should be used in their applications.

1. Introduction

With the development of information technology, amount of data are collected from various sensors and devices in multiple formats. Such data processed by independent or connected applications will routinely cross the peta-scale threshold, which would in turn increase the computational requirements. The fast increase and update of big data brings a new challenge to quickly acquire the useful information with data mining techniques. For the purpose of processing big data, Google developed a software framework called MapReduce to support large distributed data sets on clusters of computers [8,9], which is effective to analyze large amounts of data. As one of the most important cloud computing techniques, MapReduce has been a popular computing model for cloud computing platforms [49]. To extend the MapReduce to be support for iterative programs in many applications including data mining, web ranking and graph analysis, iterative MapReduce (iMapReduce) are proposed [6,11]. In addition, many MapReduce runtime systems are developed and lots of traditional

☆ This is an extended version of the paper presented at 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications, KDD2012, Beijing, 2012, China.

* Corresponding author.
E-mail addresses: JunboZhang86@163.com, jbzhang@cs.gsu.edu (J. Zhang), jwong9@student.gsu.edu (J.-S. Wong), trli@swjtu.edu.cn (T. Li), pan@cs.gsu.edu (Y. Pan).

0888-613X/$ – see front matter
http://dx.doi.org/10.1016/j.ijar.2013.08.003

methods combined with MapReduce have been presented. Here we review some implementations of MapReduce, iMapReduce, and traditional methods combined with MapReduce model.

- *Implementations of MapReduce*.
  (1) Apache Hadoop [41] was developed for data-intensive distributed applications. It is open source software framework and helps to construct the reliable, scalable, distributed systems. (2) Phoenix [37] is a shared-memory implementation of MapReduce model for data-intensive processing tasks, which can be used to program multi-core chips as well as shared-memory multiprocessors. (3) Aiming to provide a generic framework for developers to implement data- and computation-intensive tasks correctly, efficiently, and easily on the GPU, Mars [18] was developed for graphic processors (GPUs) using MapReduce framework. Mars hides the programming complexity of the GPU behind the simple and familiar MapReduce interface. Hence, the developers can write their code on the GPU without any knowledge of the graphics APIs or the GPU architecture. (4) MapReduceRoles4Azure (MR4Azure) [14] is a distributed decentralized MapReduce runtime for Windows Azure that was developed using Azure cloud infrastructure services.
- *Implementations of iMapReduce*.
  (1) Iterative MapReduce model was introduced in Twister [11], which is a lightweight MapReduce runtime system. It provides the feature for cacheable MapReduce task, which allows developer to develop iterative applications without spending much time on reading and writing large amount of data in each iteration. Another version for Windows Azure called Twister4Azure [15] also has been released. (2) HaLoop [6] is a modified version of the Hadoop MapReduce framework, designed to serve iterative applications. It does not only extend MapReduce with programming support for iterative applications, but also dramatically improves their efficiency by making the task scheduler loop-aware and by adding various caching mechanisms. (3) In addition, Microsoft also developed an iterative MapReduce runtime for Windows Azure, code-named Daytona [3], which is designed to support a wide class of data analytics and machine learning algorithms. It can scale out to hundreds of server cores for analysis of distributed data.
- *Traditional methods combined with MapReduce*.
  Apache Mahout [33] can help developers to produce implementations of scalable machine-learning algorithms on Hadoop platform. Menon et al. gave a rapid parallel genome indexing with MapReduce [31]. Blanas et al. proposed crucial implementation details of a number of well-known join strategies for log processing in MapReduce [5]. Ene et al. developed fast clustering algorithms using MapReduce with constant factor approximation guarantees [12]. Lin et al. presented three design patterns for efficient graph algorithms in MapReduce [29].

Granular computing (GrC) is an emerging information processing paradigm in computational intelligence [50]. It is a framework to create computer systems employing a human-centric view of the world [10]. Rough set theory is considered as one of the leading special cases of GrC approaches. As one of data analysis techniques, rough sets based methods have been successfully applied in data mining and knowledge discovery during last decades [13,34,45], and particularly useful for rule acquisition [22–25,40] and feature selection [19,20,27,28,35,44].

To mine knowledge from very large data sets based on rough sets, incremental techniques are employed to improve the computational efficiency [7,26,30,48]. In addition, positive approximation can be used to accelerate feature selection and rule acquisition process [21,35,36]. Susmaga presented an effective method for parallel computation of reducts with rough set theory [38]. But, to our knowledge, most of the current algorithms based on rough sets are the sequential algorithms and corresponding tools only run on a single computer to deal with small data sets. To expand the applications of rough sets in the field of data mining and knowledge discovery from big data, we proposed a parallel method for computing approximations based on rough sets and MapReduce [47]. Furthermore, a parallel method for knowledge acquisition using MapReduce was presented [46]. Based on these work, we discuss about rough set based parallel large-scale methods for knowledge acquisition in this paper. The corresponding parallel algorithms are designed for knowledge acquisition on the basis of the characteristics of the data. The proposed algorithms are implemented on several representative MapReduce runtime systems: Hadoop [41], Phoenix [39] and Twister [11]. We test the proposed algorithms on these runtime systems and compare their performance. Comprehensive experimental results demonstrate that the proposed algorithms can effectively process very large data sets.

The paper is organized as follows. Section 2 includes a background introduction to MapReduce and rough sets. Rough set based methods for knowledge acquisition with MapReduce are presented in Section 3. Experimental analysis is given in Section 4. The paper ends with conclusions in Section 5.

2. Preliminaries

In this section, we review MapReduce technique [8,9] and some basic concepts of rough sets and knowledge acquisition [30,34,40,47].

2.1. MapReduce programming model

MapReduce [8,9], by Google, is a distributed programming model for processing large-scale data, which is described as follows.