Parallel *K*-Means Clustering Based on MapReduce

Weizhong Zhao[1,2], Huifang Ma[1,2], and Qing He[1]

[1] The Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences
[2] Graduate University of Chinese Academy of Sciences
{zhaowz,mahf,heq}@ics.ict.ac.cn

Abstract. Data clustering has been received considerable attention in many applications, such as data mining, document retrieval, image segmentation and pattern classification. The enlarging volumes of information emerging by the progress of technology, makes clustering of very large scale of data a challenging task. In order to deal with the problem, many researchers try to design efficient parallel clustering algorithms. In this paper, we propose a parallel k-means clustering algorithm based on MapReduce, which is a simple yet powerful parallel programming technique. The experimental results demonstrate that the proposed algorithm can scale well and efficiently process large datasets on commodity hardware.

Keywords: Data mining; Parallel clustering; K-means; Hadoop; MapReduce.

1 Introduction

With the development of information technology, data volumes processed by many applications will routinely cross the peta-scale threshold, which would in turn increase the computational requirements. Efficient parallel clustering algorithms and implementation techniques are the key to meeting the scalability and performance requirements entailed in such scientific data analyses. So far, several researchers have proposed some parallel clustering algorithms [1,2,3]. All these parallel clustering algorithms have the following drawbacks: a) They assume that all objects can reside in main memory at the same time; b) Their parallel systems have provided restricted programming models and used the restrictions to parallelize the computation automatically. Both assumptions are prohibitive for very large datasets with millions of objects. Therefore, dataset oriented parallel clustering algorithms should be developed.

MapReduce [4,5,6,7] is a programming model and an associated implementation for processing and generating large datasets that is amenable to a broad variety of real-world tasks. Users specify the computation in terms of a *map* and a *reduce* function, and the underlying runtime system automatically parallelizes the computation across large-scale clusters of machines, handles machine failures, and schedules inter-machine communication to make efficient use of the

M.G. Jaatun, G. Zhao, and C. Rong (Eds.): CloudCom 2009, LNCS 5931, pp. 674–679, 2009.

network and disks. Google and Hadoop both provide MapReduce runtimes with fault tolerance and dynamic flexibility support [8,9].

In this paper, we adapt k-means algorithm [10] in MapReduce framework which is implemented by Hadoop to make the clustering method applicable to large scale data. By applying proper <key, value> pairs, the proposed algorithm can be parallel executed effectively. We conduct comprehensive experiments to evaluate the proposed algorithm. The results demonstrate that our algorithm can effectively deal with large scale datasets.

The rest of the paper is organized as follows. In Section 2, we present our parallel k-means algorithm based on MapReduce framework. Section 3 shows experimental results and evaluates our parallel algorithm with respect to speedup, scaleup, and sizeup. Finally, we offer our conclusions in Section 4.

2 Parallel *K*-Means Algorithm Based on MapReduce

In this section we present the main design for Parallel K-Means (PKMeans) based on MapReduce. Firstly, we give a brief overview of the k-means algorithm and analyze the parallel parts and serial parts in the algorithms. Then we explain how the necessary computations can be formalized as map and reduce operations in detail.

2.1 *K*-Means Algorithm

K-means algorithm is the most well-known and commonly used clustering method. It takes the input parameter, k, and partitions a set of n objects into k clusters so that the resulting intra-cluster similarity is high whereas the inter-cluster similarity is low. Cluster similarity is measured according to the mean value of the objects in the cluster, which can be regarded as the cluster's "center of gravity".

The algorithm proceeds as follows: Firstly, it randomly selects k objects from the whole objects which represent initial cluster centers. Each remaining object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster center. The new mean for each cluster is then calculated. This process iterates until the criterion function converges.

In k-means algorithm, the most intensive calculation to occur is the calculation of distances. In each iteration, it would require a total of (nk) distance computations where n is the number of objects and k is the number of clusters being created. It is obviously that the distance computations between one object with the centers is irrelevant to the distance computations between other objects with the corresponding centers. Therefore, distance computations between different objects with centers can be parallel executed. In each iteration, the new centers, which are used in the next iteration, should be updated. Hence the iterative procedures must be executed serially.