Equi-join for Multiple Datasets Based on Time Cost Evaluation Model

Hong Zhu, Libo Xia, Mieyi Xie*, and Ke Yan

School of Computer Science and Technology,
Huazhong University of Science and Technology,
Wuhan, Hubei, 430074, P.R. China
{zhuhong,xiemeiyi}@hust.edu.cn

Abstract. MapReduce is an important programming model for processing big data with a parallel, distributed algorithm on a cluster. In big data analytic application, equi-join is an important operation. However, it is inefficient to perform equi-join operations in MapReduce when multiple datasets are involved in the join. In this paper, a time cost evaluation model is extended for an equi-join by considering the time cost of calculation. In addition, the sub-joins in an equi-join are classified into star pattern sub-joins on single attribute and chain pattern sub-joins. Based on the extended model, optimization methods are presented and an equi-join plan with lower time cost is chosen for the equi-join. The optimization methods include: the star pattern sub-joins on one attribute are first processed; next, a chain pattern sub-join with minimal scale of intermediate results (i.e. the number of tuples in intermediate results) is processed; at last, a chain pattern sub-join is decomposed into several MapReduce jobs or single MapReduce job by dynamic programming to obtain an optimal scheme for the chain pattern sub-join. We conducted extensive experiments, and the results show that our method is more efficient than those methods such as MDMJ, Hive and Pig.

Keywords: Join, MapReduce, Dynamic Programming.

1 Introduction

With the development of information technology, massive amount of data are collected in many fields, such as medical, finance, communication, and governments. Nowadays, there are urgent needs for analyzing big data in these applications. However, solutions based on conventional distributed or parallel databases are difficult to meet the needs of big data analysis. MapReduce is an important programming model for processing big data with parallel, distributed algorithms on a cluster [3]. Nowadays, thousands of projects for big data processing have been implemented by this model, including large-scale image processing, machine learning as well as many other areas. In data analytical queries, equi-join is an important operation. However, it is not efficient to perform an equi-join

* Corresponding author.
operation in MapReduce when multiple datasets are involved in the join. Although several approaches performing an equi-join are presented in literatures [1, 5–10], these approaches have advantages for the equi-joins on some special datasets and most of them are not general on any datasets.

The approaches for equi-joins are classified into two groups: Map-side join and Reduce-side join [5]. The Broadcast-Join and its improved approaches Semi-Join and Per-Split Semi-Join are all Map-side joins [2]. However, the performances of these approaches for equi-joins decrease seriously when the scales of datasets involved in equi-joins increase, because the efficiencies of these methods depend on the hardware of the clusters. There are also several approaches for Reduce-side joins on multiple datasets, such as the equi-join method with multiple MapReduce jobs (MRJs) [7, 9], multi-dimensional Reducer matrix based multi-join (MDMJ) [1], and modifying original MapReduce frameworks [8, 10]. These methods have different advantages when specific datasets are joined. Amongst these approaches, an equi-join is processed by a series of MRJs or single MRJ. However, it is difficult to determine whether an equi-join should be processed by single MRJ or by multiple MRJs. If we are able to evaluate the time cost for disk I/O, communication and calculation of an equi-join, we could choose a plan with time cost as low as possible from different schemes of the equi-join. Then the efficiency of the equi-join can be improved. We think this is an issue.

The contributions of this paper are listed in the following.

1. The time cost of calculation for an equi-join is extended based on the time cost model in literature [11]. Therefore, time cost for an equi-join consists of three parts: disk I/O, communication, and calculation.

2. Based on the time cost model, optimization methods are presented and an equi-join plan with lower time cost is chosen. Then the performance of the equi-join is improved. The optimization methods include: the star pattern sub-joins on one attribute are first processed; next, a chain pattern sub-join with minimal scale of intermediate results (the number of tuples in intermediate results) is processed; at last, an optimal plan for the chain pattern sub-join is obtained by dynamic programming.

3. We conduct extensive experiments to verify the efficiency of our method. Experimental results show that the performance of our approach is better than that of other methods such as MDMJ, Hive and Pig.

The rest of this paper is organized as follows. Related work is briefly introduced in Section 2. The extended cost model is illustrated in Section 3. The optimization methods for equi-joins are presented in Section 4. In Section 5, experiments are illustrated and at last in Section 6 we summarize the conclusion.

2 Related Work

2.1 Equi-join on Single Attribute

Equi-join on single attribute is an equi-join based on multiple datasets and one attribute. For example, $R_1(a, b_1) \bowtie R_2(a, b_2) \bowtie \cdots \bowtie R_L(a, b_L)$ is a typical equi-join on one attribute $a$. During Map phase, the Map() function produces