A Predictive Map Task Scheduler for Optimizing Data Locality in MapReduce Clusters

Mohamed Merabet, EEDIS Laboratory, DjillaliLiabes University of Sidi Bel Abbes, Sidi Bel Abbes, Algeria
Sidi mohamed Benslimane, LabRI Laboratory, Ecole Superieure en Informatique, Sidi Bel-Abbes, Algeria
Mahmoud Barhamgi, Claude Bernard Lyon 1 University, Lyon, France
Christine Bonnet, Université Lyon 1, Lyon, France

ABSTRACT

This article describes how data locality is becoming one of the most critical factors to affect performance of MapReduce clusters because of network bisection bandwidth becomes a bottleneck. Task scheduler assigns the most appropriate map tasks to nodes. If map tasks are scheduled to nodes without input data, these tasks will issue remote I/O operations to copy the data to local nodes that decrease execution time of map tasks. In that case, prefetching mechanism can be useful to preload the needed input data before tasks is launching. Therefore, the key challenge is how this article can accurately predict the execution time of map tasks to be able to use data prefetching effectively without any data access delay. In this article, it is proposed that a Predictive Map Task Scheduler assigns the most suitable map tasks to nodes ahead of time. Following this, a linear regression model is used for prediction and data locality based algorithm for tasks scheduling. The experimental results show that the method can greatly improve both data locality and execution time of map tasks.

KEYWORDS

Data Locality, Execution Time Prediction, Map Task Scheduling, MapReduce, Prefetching

1. INTRODUCTION

The increasing amount of data generated by commercial and scientific applications such as social networks, scientific research, and recently Internet of Things, has become an important and challenging problem (Min et. al., 2014; Marcos et. al., 2015). New scalable programming paradigms and complex scheduling algorithms for efficiently processing such big data applications are a necessity for achieving good performance.

Hadoop, an open source implementation of the MapReduce model, has emerged as one of the most used tools, due to its extremely easy to use program, fast speed, scalability, and fault-tolerance. Several companies such as Google, Facebook, Microsoft, IBM, Amazon and many others have started using Hadoop for processing large-scale data volumes in moderate time.

Data locality is an important factor impacting the efficiency of task scheduling (Ching-Hsien et. al., 2015; Shabeera et. al., 2015). In Hadoop, data are distributed and stored locally on nodes. Tasks are also deployed to all nodes independently form data. To execute a map task on a node without local data inputs, the node needs to transfer data from remote data providers, which delays the execution of...
tasks. In order to improve the data locality, Hadoop tries to decrease the impact of data transmission by scheduling tasks preferentially to nodes with corresponding input data.

Data prefetching has received considerable attention in the literature as an interesting mechanism of boosting the performance in distributed systems (Sun et al., 2014; Sun et al., 2016). It hides data access latency by overlapping data transfer tasks with data processing. However, the problem is to determine what and when to pre-fetch for achieving a good pre-fetching accuracy.

In Hadoop, a map task may be seen as a sequence of data processing and data transmission procedures. That means when the data is processed in a map task, no data transmission is performed. As a result, data processing has to wait for the next data slice to be transferred. An improvement by using a data prefetching mechanism is proposed.

In this paper, a predictive scheduler with data prefetching is proposed to improve the data locality in MapReduce Clusters. This paper is organized as follows. Section 2 describes the architecture of MapReduce. Section 3 presents related work on locality based scheduling methods in Hadoop. In section 4, we introduce our Predictive Map Task scheduler. In section 5, we present the followed steps to evaluate the implemented schedule. In section 6, we discuss the results obtained during evaluations. Finally, conclusions and future work are given in Section 7.

2. ARCHITECTURE OF MAPREDUCE

MapReduce is a programming model for large-scale data-intensive distributed data processing. It divides the execution in two phases: map and reduce. In the map phase, amounts of map tasks process data blocks independently. After all map tasks are finished, the reduce phase begins. The intermediate results of map tasks are shuffled, sorted, and processed in parallel with one or more reduce tasks.

A user submits a job comprising of a map function and a reduce function which are subsequently transformed into map and reduce tasks scheduled on slots hosted by participating nodes in the cluster. HDFS loads data partitions into fixed equal-size splits, and distributes splits across cluster nodes. Each split is assigned a map task.

The master node in MapReduce is referred to as Job Tracker. Each slave node is denoted as Task Tracker. Job Tracker and Task Trackers communicate over the cluster network via a heartbeat mechanism. Hadoop’s framework adopts a pull scheduling strategy rather than a push one. That is, Job Tracker does not push map and reduce tasks to Task Trackers, but rather Task Trackers pull them by making pertaining requests. Every Task Tracker sends a heartbeat message periodically to Job Tracker encompassing a request for a map or a reduce task to run. Job Tracker satisfies requests for map tasks via attempting to schedule mappers in the vicinity of their input splits.

3. RELATED WORK

Research on improving MapReduce scheduling techniques has recently received an important attention. In this section, we review the work related to scheduling algorithms that exploit the data locality using different heuristics and mathematical methods.

3.1. Job-Level Approaches

(Zaharia et al., 2010) introduce a delay-scheduling algorithm to improve the locality of the data. When a node requests a particular task, if the current job cannot assign a local task, the scheduler skips this task and looks for next jobs. However, if a job has been skipped long enough, the scheduler will allow it to initiate non-local tasks, to avoid starvation. This method improves the locality problem by causing the work to wait for a scheduling opportunity on a node with local data. Therefore, it optimizes only map tasks with local data. In our method, we optimize all map tasks.
Structural Outlooks for the OTIS-Arrangement Network
Ahmad Awwad, Jehad Al-Sadi, Bassam Haddad and Ahmad Kayed (2011).
International Journal of Grid and High Performance Computing (pp. 59-68).
www.igi-global.com/article/structural-outlooks-otis-arrangement-network/54196?camid=4v1a

A Semantic-Driven Adaptive Architecture for Large Scale P2P Networks
www.igi-global.com/article/semantic-driven-adaptive-architecture-large/47208?camid=4v1a