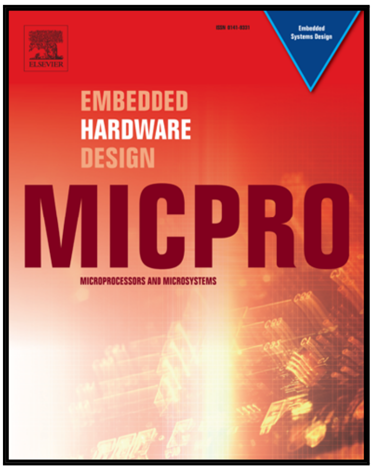

MapReduce Short Jobs Optimization Based On Resource Reuse

Yuliang Shi , Kaihui Zhang , Lizhen Cui , Lei Liu ,
Yongqing Zheng , Shidong Zhang , Han Yu

PII: S0141-9331(16)30043-6
DOI: 10.1016/j.micpro.2016.05.007
Reference: MICPRO 2397

To appear in: *Microprocessors and Microsystems*

Received date: 13 January 2016
Revised date: 18 April 2016

Please cite this article as: Yuliang Shi , Kaihui Zhang , Lizhen Cui , Lei Liu , Yongqing Zheng , Shidong Zhang , Han Yu , MapReduce Short Jobs Optimization Based On Resource Reuse, *Microprocessors and Microsystems* (2016), doi: 10.1016/j.micpro.2016.05.007

MapReduce Short Jobs Optimization Based On Resource Reuse

Yuliang Shi[*, a], Kaihui Zhang[a], Lizhen Cui[a], Lei Liu[a], Yongqing Zheng[a], Shidong Zhang[a], Han Yu[b]
[a]Shandong University,Jinan, China
[b]Nanyang Technological University, Singapore, Singapore

Abstract—**Hadoop is an open-source implementation of MapReduce serving for processing large datasets in a massively parallel manner. It was designed aiming at executing large-scale jobs in an enormous number of computing nodes offering computing and storage. However, Hadoop is frequently employed to process short jobs. In practice, short jobs suffer from poor response time and run inefficiently. To fill this gap, this paper analyses the process of job execution and depicts the existing issues why short jobs run inefficiently in Hadoop. According to the characteristic of task execution in multi-wave under cluster overload, we develop a mechanism in light of resource reuse to optimize short jobs execution. This mechanism can reduce the frequency of resource allocation and recovery. Experimental results suggest that the developed mechanism based on resource reuse is able to improve effectiveness of the resource utilization. In addition, the runtime of short jobs can be significantly reduced.**

Keywords—Hadoop; short job; performance optimization; resource utilization; task scheduling.

1. Introduction

Numbers of enterprises, financial institutions and media organizations are under the pressure of processing large-scale datasets, however, conventional data processing tools and computing models do not handle it. Hadoop which is an open-source implementation of MapReduce [1] proposed by Google provides an effective solution to handle large-scale datasets. MapReduce Jobs submitted to Hadoop are divided into Map tasks and Reduce tasks which run in a massively parallel manner on multiple nodes, so that the runtimes of jobs are reduced significantly. Hadoop hides many details of parallel computing, such as distributing data blocks to computing nodes, rerunning failed tasks, and makes users focus on specific business logic processing. Moreover, Hadoop provides a good scalability, high-availability and fault tolerance, which make Hadoop become the mainstream computing framework to run data-intensive and compute-intensive applications. So the academic community begins to pay close attention to Hadoop and deals with many problems, such as unfairness [2, 3, 4, 5], stragglers [4, 6, 7], and data skew [8, 9, 10, 11].

Hadoop was originally designed for long-run jobs on a large number of computing nodes, but it is often used to handle short jobs in practice. The runtime of short job is less than a specified threshold set by users. Short jobs can be distinguished from long jobs by the size of input datasets, the number of tasks job divided, the resource required by task, the runtime of task and the runtime users expect. Since Hadoop does not take the characteristics of short jobs into account, short jobs run inefficiently.

The hardware configurations of nodes in a cluster, job scheduling algorithm and cluster load are crucial factors affecting job performance. When scheduling task, Hadoop assumes that nodes in a cluster are homogeneous. With the gradual expansion of a cluster scale, however, hardware configurations of new nodes which are added to a cluster are significantly higher than the configurations of old ones. Therefore, tasks jobs split run efficiently on new nodes compared with on old nodes. In the case of cluster heavy load, tasks which jobs split cannot obtain sufficient resources immediately to run and a part of tasks are put into a waiting queue. The running tasks release the occupied resources when tasks finish. Hadoop picks up an appropriate task from waiting queue and assigns available resources to it according to scheduling algorithm specified by user. So, if the amount of resources which tasks request exceeds the amount of available resources a cluster offers, tasks are executed in multiple waves. In TaoBao’s Hadoop cluster, over 70% of Map tasks run more than two waves. Therefore, cluster load has a decisive influence on the response time and runtime of a job.

The purpose of this paper is to improve the execution performance of short jobs. This paper analyzes the execution process of jobs and describes the disadvantages of executing short jobs in Hadoop cluster. According to the characteristics of task running in multiple waves under cluster heavy load, we develop an optimization mechanism based on resources reuse. First, we add a kind of statistic heartbeat which contains the statistic data of running task. Second, we build task performance model to predict the runtime of task. Finally, based on task performance model, we design a task sub-scheduler which can make the unscheduled task reuse the resources released by the running task. Because the number of Map tasks executed in multiple waves is more than the number of Reduce tasks executed in multiple waves, we only optimize Map tasks jobs divided in this paper. Experimental results suggest that optimization method can effectively reduce the runtime of short jobs and significantly improves the resource utilization of cluster.

This paper is organized as follows. Related work is discussed in Section 2. Section 3 presents the preliminary of task execution. The design and implementation of developing optimization mechanism are given in section 4. Section 5 evaluates the developed optimization mechanism. Finally, we conclude the paper.

* Corresponding author
E-mail addresses: shiyuliang@sdu.edu.cn, kaihuizhang@126.com,clz@sdu.edu.cn,l.liu@sdu.edu.cn,
zhengyongqing@dareway.com.cn,zsd@sdu.edu.cn,han.yu@ntu.edu.sg