ISSN 2395-1621

Implementation of Dynamic Slot Configurations for Homogeneous and Heterogeneous Hadoop Clusters

#1Prof.G.V.Kadam , #2Archana Dirgule

1archana.dirgule@gmail.com

JSPM's Rajarshi Shahu School of Engineering & Research,Narhe, Pune, Maharashtra, India.

ABSTRACT

Now a days MapReduce becomes very popular technology for large data centers and clusters. However, due to the pre-allocation of slots among map and reduce tasks, and the rigid execution order between map and reduce tasks in a MapReduce environment, slots can be severely under-utilized, which significantly degrades the performance .We realized that this kind of static configuration may affect negatively on system resource utilizations as well as long completion length. So we have proposed simple and effective schemes which utilizes slot ratio between map and reduce tasks as a tunable knob for reducing the makespan of a given set.

Keywords: MapReduce, slot allocation, Hadoop fair scheduler, TaskTracker, makespan, slot pre scheduling.

ARTICLE INFO

Article History

Received : 15th June 2016

Received in revised form :

15th June 2016

Accepted : 18th June 2016

Published online :

18th June 2016

I. INTRODUCTION

NOW a days, MapReduce has become a popular parallel computing paradigm for large scale data processing in clusters and data centers. Hadoop is the open source implementation of MapReduce has been used to deploy in large clusters which contains thousands of machines used by companies like Yahoo! and Facebook to support for large jobs submitted from users. Hadoop has a static slot configuration, which means a fixed number of map slots and reduce slots which are only used for processing map reduce tasks. Map tasks can run by map slots, and reduce tasks can run in reduce slots. This static slot configuration may lead to poor performance and low resource utilization. Apache Hadoop components are responsible for running large data sets. Essential components for parallel processing Hadoop are Hadoop Distributed File System (HDFS), Hadoop YARN, and Hadoop MapReduce. In some of traditional approaches the resource utilization problem still exists which are solved by using dynamic split model of resource utilization. Considering the load in the cluster and all jobs status in run time the resources are allocated. Under this condition resource usage pipeline is used. It uses buffer for the enlargement in map phase and requirement of slots for map task and reduce tasks. It shows markable gain in performance. Dynamic slot configuration is one of the important factors while processing a large data set with MapReduce paradigm. Examined to be an evaluation over the dynamic slot configuration and scheduling techniques for MapReduce cluster. Due to this, we propose simple and effective schemes which use slot ratio between map and reduce tasks as a tunable knob for reducing the makespan of a given set. By improving the workload information of recently completed jobs, our schemes dynamically allocates resources (or slots) to map and reduce tasks.

II. LITERATURE SURVEY

Existing work focus on the full slot utilization for map while reduce slots are blank and vice versa. Hence they are cruelly under utilized. They also consider the speculative execution efficiency to be high in the single job only. They did not consider the cluster efficiency in the above consideration. The data locality maximization is important for the slot utilization efficiency and MapReduce workloads performance improvement. But there is conflict between the fairness and data locality optimization.

A. Verma, L. Cherkasova, R. Campbell. offered Scheduling task which consider the specific constraint which definitely improve the overall development of the system. Here scheduling of task is carried out considering the data as the main feature for scheduling. Data location is identified and highly required data for computation are collected along with size and location. This information is precious while scheduling the tasks.

This methodology keeps basic level information associated with node. This provides foundation to reach desired performance.

B. The authors in the paper are Z. H .Guo, et al. explored that resource utilization problems which are still faced by some of traditional approaches which are solved by using dynamic split model of resource utilization. Considering the load in the cluster and all jobs status in run time the resources are allocated. Under this condition resource usage pipeline is used. It uses buffer for the enlargement in map phase and requirement of slots for map task and reduce tasks. It shows markable gain in performance

C. The authors Yu S. Tan, et al. proposed a system for Map reduce clusters in Hadoop which is one of the famous deployments of cluster. But in the darker side, most map reduce implementations are designed and used for homogenous clusters which furnish low level performance on heterogeneous clusters. This paper indentifies the aspects such as system configuration and task scheduling with respect to different scenario in configuration and scheduling to uplift the current features. Benefit is that every shuffle operation is effective and has positive impact on performance. Dark side is that in early stage of shuffle resource utilization might be complex.

D. J. Wolf et al. introduced FLEX, which is an add-on module integrated with HFS. Because FIFO scheduling in MapReduce causes job starvation. Hadoop Fair scheduler (HFS) implemented for achieving a degree of fairness. The goal is to optimize scheduling metrics such as completion time length, response time, stretch, and Service Level Agreements.

E. MROrchestrator, means a MapReduce resource Orchestrator framework, which dynamically identifies resource bottlenecks, and resolve them with coordinated, fine-grained and on demand resource allocations. They have implemented MROrchestrator on two 24-node natives and virtualized Hadoop clusters.
Experimental results with a suite of representative MapReduce benchmarks demonstrate up to 38 percent reduction in job completion times, and up to 25 percent increase in resource utilization

III. DYNAMIC SLOT CONFIGURATION

System Architecture:

The system architecture Figure 1 gives an overview of three slot allocation techniques, i.e., Slot Allocation, Speculative Execution Performance Balancing and Slots prescheduling.
The main objective of the proposed system
is to optimize the resource allocation first using
dynamic Hadoop slot allocation, second speculative execution performance balancing and last is. slot prescheduling for classification operation using medical dataset. Dynamic SC consists of three optimization techniques, namely,

i)Dynamic Hadoop Slot Allocation (DHSA),

_ Speculative Execution Performance Balancing
(SEPB)
ii)SlotPreScheduling.

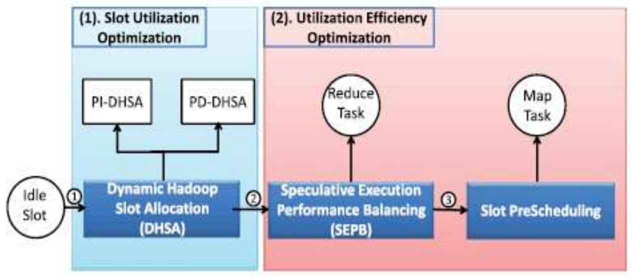

(1). Slot Utilization Optimization
(2). Utilization Efficiency Optimization
PI-DHSA
PD-DHSA
Reduce Task
Map Task
Idle Slot
①
Dynamic Hadoop Slot Allocation (DHSA)
②
Speculative Execution Performance Balancing (SEPB)
③
Slot PreScheduling

Fig 1: System Architecture

Dynamic SC will first attempt to improve the slot utilization with DHSA. Decisions are taken dynamically for allocation based on constraints. If the allocation is true, Dynamic MR will further optimize the performance by improving the efficiency of slot utilization with SEPB. Dynamic SC will be able to further improve the slot utilization efficiency from the data locality optimization aspect with Slot Prescheduling. Allocated slots are generic and can be used by either map or reduce tasks, although there is a pre-configuration for the number of map and reduce slots. For processing the cancer data we should allocate slots. For that here we use dynamic Hadoop slot allocation. Our dynamic slot configuration system is based on the observation which has been taken at different time period there may be idle map (or reduce) slots, as the job proceeds from map phase to reduce phase. If insufficient map slots are present, it use all the map slots and borrow slots from the reduce slots and vice versa. Statically map tasks uses map slots and likewise reduce tasks prefer to use reduce slots. The benefit is that, the pre-configuration of map and reduce slots per slave node can still work to control the ratio of running map and reduce tasks during runtime. Fairness is an important metric in Hadoop Fair Schedule. It contains two alternatives, namely, pool independent DHSA(PI-DHSA)and pool-dependent DHSA (PDDHSA),each of which considers the fairness from different aspects. We have the following three observations. Firstly, the original Hadoop is very sensitive to the map/reduce slot configuration, whereas there is little impact for the map/reduce slot configuration on our DHSA (i.e., the speedup keeps stable under different map/reduce slot configurations). For example, there are about1.8x performance differences for Sort benchmark between the optimal and worst-case map/reduce slot configurations for the original Hadoop. To explain the reason behind it, let's take a single job for example. Let N_M and N_R denote the number of map tasks and reduce tasks. Let t_M And t_R denote the execution time for a single map task and reduce task. Let S_M and S_R denote the number of map slots and reduce slots. Moreover, we assume that there is one slot per CPU core and thus the sum of map slots and reduce slots is fixed for a given cluster. Then for the traditional Hadoop cluster, the execution time will be

$$T_{EXE} = \left\lceil \frac{N_M}{S_M} \right\rceil . t_M + \left\lceil \frac{N_R}{S_R} \right\rceil . t_R .$$

In contrast, it will be

$T_{EXE} = \left\lceil \frac{N_M}{S_M + S_R} \right\rceil \cdot t_M + \left\lceil \frac{N_R}{S_M + S_R} \right\rceil \cdot t_R$ for our DHSA. Based on the formula, we can see varied performance from the traditional Hadoop under different slot configurations. However, there is little impact on the performance for different slot configurations under DHSA

Map Reduce jobs execution time is very sensitive to slow running task i.e. straggler. We propose a dynamic task allocation mechanism called SPEB i.e. Speculative Execution Performance Balancing for a batch of jobs with speculative execution tasks on top of Hadoop's current task selection policy. Hadoop chooses a task from a job based on the following priority: first, any failed task is given the highest priority. Second, the pending or remaining tasks are considered. For map, tasks with data local to the compute node are chosen first. Finally, Hadoop looks for a straggling task to execute speculatively. To improve data locality, we propose a Slot PreScheduling technique that can improve the data locality while having no negative impact on the fairness of MapReduce jobs. In contrast to delay scheduler, it is achieved at the expense of load balance across slave nodes.

IV. ALGORITHM

Slot Allocation-based Approximation Algorithm:

To improve the scalability and efficiency, propose a slot allocation-based approximation solution to further speed up solutions for large scale problems. This algorithm works on three cases i.e
Case 1. The map tasks which are running on map slots and reduce tasks are run on reduce slots, There is no borrowing of map and reduce slots.

Case 2. We satisfy reduce tasks for reduce slots first and then use those idle reduce slots for running map tasks.

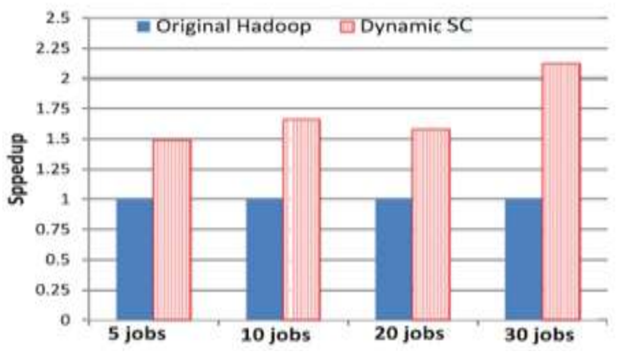
Case 3. We can schedule those unused map slots for running reduce tasks.

Case 4. The system should be in completely busy state.

V. RESULT ANALYSIS

Performance Improvement For Dynamic Slot Configuration

For the original Hadoop, we choose the optimal slot configuration for MapReduce jobs one by one all the possible slot configurations. Our aim is to compare the performance for Dynamic Slot Configuration with the original Hadoop under the optimal map/reduce slot configuration for MapReduce jobs. Fig. 2 presents the evaluation results for a single MapReduce job as well as MapReduce workloads consisting of multiple jobs. Particularly, for multiple jobs, we consider 5 jobs, 10 jobs, 20 jobs, and 30 jobs.



VI. CONCLUSION AND FUTURE WORK

All observations are calculated with respect to the original Hadoop. We can see that, even under the optimized map/reduce slot configuration for the original Hadoop, our DynamicSC system can still further improve the performance of MapReduce jobs significantly, i.e., there are about 46 -115 percent for a single job and 49 -112 percent for MapReduce workloads with multiple jobs. Inefficienctmap reduce identification and dealing is considered as major issue. Slot where dynamically map and reduce tasks performance are used to reach high rate in performance and assignment of tasks to slots are done dynamically. To achieve efficiency for tasks performed in slots which is crucial. Speculative execution performance balance used to raise performance of group of jobs and prescheduling used to achieve data locality at high rate overcome loop holes and do the required operation in positive direction. All of these features help to uplift the implementation of map reduce. Future work will be able to consider implementing DynamicSC for cloud computing environment with more metrics (e.g., budget, deadline) considered and different platforms by reviewing some existing works.

REFERENCES

[1] J. Dean, and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," in Proc.6th conf. Symp. Operating Syst. Des.
Implementation, 2004, pp. 107–113.

[2] Hadoop, http://hadoop.apache.org

[3] Z. H. Guo, G. Fox, M. Zhou, and Y. Ruan, "Improving resource utilization in mapreduce," in Proc. IEEE Int. Conf. Cluster Comput.,2012, pp. 402–410.

[4] Yi Yao Jiayin Wang Bo Sheng Chiu C. Tan Ningfang"Self- Adjusting Slot Configurations for Homogeneous and Heterogeneous Hadoop Clusters".

[5] . A. Verma, L. Cherkasova, R. Campbell. "Two Sides of a Coin: Optimizing the Schedule of MapReduce Jobs to Minimize Their Makespan and Improve Cluster Performance. In IEEE MASCOTS, pp. 11-18, 2012. "

[6] Yu S. Tan, Bu-Sung Lee, Bingsheng He and Roy H. Campbell. "A Map-Reduce Based Framework for Heterogeneous Processing Element Cluster Environments. In Proc. of CCGRID 2012"

[7] Joel Wolf, Deepak Rajan, Kirsten Hildrum, Rohit Khandekar, Vibhore Kumar, Sujay Parekh, Kun-Lung Wu, Andrey Balmin “FLEX: A Slot Allocation Scheduling Optimizer for MapReduce Workloads “

[8] Bikash Sharma, RamyaPrabhakar, “MROrchestrator: A Fine-Grained Resource Orchestration Framework for MapReduce
Clusters.”