A Survey on Self Adjusting Slots & Dynamic Job Ordering for Mapreduce Workloads using Homogeneous and Heterogeneous Hadoop Cluster

Kaveri Hiremath
M-Tech Student, Dept. of CS & Engg
Dayananda Sagar University,
Bangalore, India

Dr. Reeja S. R
Associate Professor, Dept. of CS & Engg
Dayananda Sagar University,
Bangalore, India

***Abstract*— In recent years, Hadoop has became the best platform for information analysis on huge data sets. MapReduce is the simple and standard programming language which is used for BigData applications. Open source apache Hadoop and its MapReduce framework are used. Currently, Hadoop Cluster uses only fixed slot configuration, means fixed number of Mapper slots and Reducer slots throughout the lifetime of the cluster. This fixed slot configuration leads to poor performance. And also MapReduce workloads generally contain a group of jobs in which it consists of multiple Mapper tasks followed by multiple Reducer tasks. The Mapper tasks will only run in the defined Map slots and Reducer tasks will only run in the defined Reduce slots. The general execution contains Mapper tasks are executed before reducer tasks. It is observed that for different MapReduce slot configuration and execution of job orders for MapReduce workloads have extremely distinct performance and system utilization. In this paper, developed a technique called dynamically allocates the slots for MapReduce jobs and extending this project by adding job ordering optimization for MapReduce workloads in the slot configuration for better performance.**

In this survey, we discussed mapreduce slots, about reducing completion length, Hadoop performance and resource utilization.

Keywords*— *MapReduce; Hadoop; Scheduling Algorithms; Slot Allocation; Workloads; Makespan.

I. INTRODUCTION

In cloud environment, cloud scheduler plays a major role in distributing resources for various jobs execution. For processing Big-data applications, simple and standard programming model is used in the Hadoop cluster. Many organizations, researchers are running MapReduce [1] applications on cloud, now a day's.

Scheduling [4] the jobs in a MapReduce plays a major role in MapReduce applications for improving performance. The default scheduler in Hadoop MapReduce is FIFO scheduler, default scheduler in Facebook is Fair scheduler and Yahoo is Capacity scheduler. All these default schedulers are the examples for MapReduce applications which suits for static clusters. Therefore, need of dynamic scheduler which can allocate MapReduce applications based on the features of the applications.

Make span and Completion time are the two important keywords. Generally, Make span is the time taken from the starting of the job until the completion of the last job and Completion time is the total sum of completed time periods for all the jobs.

In this paper, proposes a job ordering algorithm i.e. Johnson's rule-based algorithm is used to enhance both makespan and total completion time at a same time. And also allocates the slots dynamically for MapReduce jobs using Slot Assigner algorithm.

II. MOTIVATION

Based on the challenges of the Hadoop cluster, Discussed proposed algorithm.

A. *Optimization of job ordering for resource utilization.*

The critical issue is to increase the performance of the system in Hadoop cluster. For make span optimizing of job ordering, used Johnson's rule-based algorithm. New version of Johnson's rule-based algorithm called bi-criteria heuristic algorithm is used to optimize both makespan and total completion time at the same time.

B. *Scheduling tasks and resource utilization and the job heterogeneity.*

In this part we used scheduling algorithm, which gives a result of equal distribution of resources among MapReduce workloads in the system and also maximizes the resource utilization and output in multi-user cluster environment. In this algorithm, it considers the heterogeneity and MapReduce allocation of resources in the tasks.

III. HISTORY OF MAPREDUCE AND HADOOP

In the year 2004, Google has published a paper [1] MapReduce. In this paper the proposed algorithm has solved problems like parallelization, Distribution, Fault-tolerance.

In 2004, for the first time MapReduce was familiarized as a programming model by Jeffery Dean & Sanjay Ghemawat of Google. They discussed Google's approach to gathering and analyzing website data for search optimization. Google's MapReduce system ran on the Google File System (GFS).

The name Hadoop began in Apache Lucern project, which it gives result as a subproject in the Apache Lucern project as a subset of that project, which it gives the result of text search in large databases. In 2006, Doug cutting, an employee of Yahoo! After seeing his son's toy elephant, named as Designed Hadoop and he released Hadoop as an open source Apache project in 2007. Using Hadoop, experimental 4000 nodes clusters was created in 2008.

IV. LITERATURE SURVEY

In Literature survey, there is research study on performance optimization of map reduce workloads.

To generate a collection of tasks, users defines a Mapper function as a key/value pair of intermediate values and a Reduce function that combines all these intermediate values associated with the same intermediate key. For many different purposes, Google has been successfully used the MapReduce [1] programming model. For the success of this model, they indicated to several reasons. This MapReduce programming model is simple/easy to improve the resources of a huge distributed system for programmers without having any knowledge in parallel and distributed systems, since it dissemble the details of parallelization, fault-tolerance, and locality optimization and load balancing. This MapReduce implementation runs on a large chunk of belonging machines and is highly extensible. Developed hundreds of MapReduce programs and MapReduce jobs are executed more than thousands every day on Google's Cluster.

Large-Scale MapReduce [1] Clusters that frequently processes a number of unstructured & semi-structured data in the clouds. To increase the effectiveness or utilization of these MapReduce clusters is the main key Challenge.

[2] Considers a small set to produce workload that consists of MapReduce tasks. By these subsets, they came to know that the order in which these tasks are executed can have a remarkable impact on their overall completion time and the resource utilization. To address this issue, they thought to robotize the design of a MapReduce job schedule that improve the completion time. They used abstraction framework called Balanced Pools to automate this design; this framework efficiently increases the performance of MapReduce jobs in a given workload for creating an optimized job schedule. By simulation, they achieved 15%-38% utilization improvements.

Quincy [3] tells about the scheduling the parallel jobs on MapReduce clusters. In this paper, for scheduling the distributed jobs, they have introduced a extendable new framework called fine-grain resource sharing . The scheduling issue is mapped to graph data structure to demands of data locality, fairness & starvation-freedom to schedule according to a global cost model.

They evaluate their implementation of this new framework called Quincy [3] on a number of computers using various workloads of data and CPU-intensive jobs. And also evaluate against actual queue based algorithm and develop several rules for each scheduler with & without fairness constraints. This proposed framework gets good propriety when fairness is requested, while effectively improving data locality. The amount of data transferred on cluster while scheduling is reduced up to the factor of 3.9 and throughput increases to 40% by the experiments.

MapReduce [1] is a programming model which is used for data analytics. They consider a multiple data analytics applications can share the same physical resources in the environment of managing MapReduce applications. This sharing of multiple applications with the data center leads to more solid workloads to save the cost and energy. In this shared environment, it is important to maintain the performance of MapReduce workloads. To address this issue they introduced a task scheduler [4] for MapReduce framework. This task scheduler allows performance-driven [4] management of MapReduce tasks. This proposed task scheduler adjusts allocation for the MapReduce jobs and dynamically estimates the performance of multiple jobs at a same time in the MapReduce tasks.

Paper [5] tells about the performance and resource utilization of the MapReduce in the multi-job workloads. Existing MapReduce schedulers defines a static number of MapReduce slots to represent the quantity of a Hadoop Cluster. This works for homogeneous workloads, but not for different resources i.e. heterogeneous workloads. Their technique holds job writing information to adjust the slots for MapReduce dynamically and also to increase the performance of the cluster.

Some classical MapReduce platforms have some low performance in the resource utilization, since the traditional multi-phase parallel model. To address this problem, they used two technologies [6] i.e. Dynamic Resource Allocation & Resource Usage Pipeline. Dynamic resource allocation considers the priority of the jobs and requirement of jobs while resource allocation and Resource Usage Pipeline assigns tasks dynamically. By using these techniques they improved their throughput by 21.72%.

The key challenge in MapReduce cluster is to control the MapReduce allocations for different applications to improve the performance. The issue is that for Mapper and Reducer jobs there is no job scheduler in a given ccompletion time, could distribute the relevant number of resources to the MapReduce jobs so that it meets the required Service Level Objective (SLO). To solve this problem, they introduced a framework, ARIA [7]. This framework is based on the observation, firstly they can profile a job which runs routinely and then uses its profile in the MapReduce performance to calculate the chunk of resource enforced to meet a deadline.

These Resource requirements are causing to bring the SLO-scheduler. In this job, ordering is formidable by the EDF (Earliest Deadline First) [7] scheduling which is a supporting policy for real-time processing. The EDF scheduling, as compared to traditional assumptions there is some interesting contrast in MapReduce environments. This creates new opportunities for various scheduling policies.

Allocating a MapReduce cluster between users is inspirable because it empowers statistical lowering cost and allows a user to share a large data set. They find that the traditional scheduling algorithms can perform poorly in MapReduce due to the characters of the MapReduce settings: available of data location and the dependence between Mapper and Reducer tasks. To overcome these problems, designed a fair [8] scheduler for MapReduce and they developed two simple techniques i.e. Delay Scheduling and Copy-Compute Splitting. By using these techniques, improves the throughput and completion time by 2 to 10.

Dynamic load balancing [9] approach to distributed server farm system. In this paper, they proposed a load balancing approach which effectively overcomes the load imbalance caused by the non-considerable number of large tasks. This proposed approach selects a set of active servers dynamically and allocates the loads in a given time. A new approach deals with very large tasks to reduce the waiting time at server queue. A task deadline can be defined as another parameter to the task priority. This priority calculates dynamically for each task in the server queue.

As we discussed in the above papers, most of the problem is to increase the performance and resource utilization and efficient resource management [10] in large data. Hadoop cluster allocates the fixed size for MapReduce tasks and static slots for MapReduce nodes. To overcome this problem, they design and implemented fine-grained, dynamic & coordinate resource management framework called MROrchestrator [10]. By using this framework they reduce 38% of job completion times and up to 25% increase in the resource utilization.

V. CONCLUSION

Now a day's, huge increases in volumes of data and big data. The main key challenge is to maintain this large amount of data, so this became an important point to research. In this paper, we discussed the method TuMM which dynamically controls the MapReduce slots for MapReduce jobs by using Slot Assigner algorithm. The scheduling algorithm is used to maximize the resource utilization and also discussed the job ordering algorithm called Johnson's rule-based algorithm to improve the make span and completion time. In the future work, consider Map/Reduce slot configuration issues for online jobs under FIFO scheduling and also for further, can consider other scheduling algorithms to compare the results to increase the performance of the system.

REFERENCES

[1] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters", Communications of the ACM, vol.51,no.1, pp.107-113,2008.

[2] A. Verma, L. Cherkasova, and R. H. Campbell, "Two sides of a coin: Optimizing the schedule of map reduce jobs to minimize their makespan and improve cluster performance", in MASCOTS'12,Aug 2012.

[3] M. Isard, Vijayan prabhakaran, J. currey et al., "Quincy: Fair scheduling for distributed computing clusters", in SOSP'09,2009, pp.261-276.

[4] J.polo, D. Carrera, Y. Becerra et al., "Performance-driven task co-scheduling for map reduce environments", in NOMS'10.2010.

[5] J. polo, C. Castillo, D. Carrera et al., "Resource-aware adaptive scheduling for map reduce clusters", in proceedings of the 12th ACM/IFIP/USENIX international conference on Middleware, 2011.

[6] X. W. Wang, J. Zhang, H. M. Liao, and L. Zha, "Dynamic split model of resource utilization in map reduce", in DataCloud-SC'11,2011.

[7] A. Verma, Ludmila Cherkasova, and R. H. Campbell, "ARIA: Automatic Resource Inference and Allocation for map reduce environments", in ICAC'11,2011, pp.235-244.

[8] M. Zaharia, D.Borthakur, J. S. Sarma et al., "Job Scheduling for multi-user map reduce clusters", University of California, Berkeley, tech.Rep., Apr.2009.

[9] L.Tan and Z. Tari, "Dynamic task assignment in server farms: Better performance by task grouping", in proc. of the Intl. Symp. On computers and communications (ISCC),2002.

[10] B. Sharma, R. Prabhakar, S.-H. Lim et al., "Mrorchestrator: A fine-grained resource orchestration framework for map reduce clusters", in CLOUD'12,2012.

[11] Hadoop. Capacity Scheduler. http://hadoop.apache.org/common/docs/r0.19.2/capacityscheduler.html.

[12] Hadoop. Fair Scheduler. http://hadoop.apache.org/common/docs/r0.20.2/fairscheduler.html

[13] TPC-H benchmark.[online].Available: https://www.tpc.org/tpch/

[14] TPC-H benchmark on pig.[online]. Available: https://issues.apache.org/jira/browse/PIG-2397