

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Website: www.ijirset.com

Vol. 6, Issue 7, July 2017

Dynamic Job Ordering and Slot Configuration Approach on Single Node System

Sameer M Khupse, Prof.Himanshu U Joshi

Department of Computer Engineering, Imperial College of Engg and Research, Pune, India

ABSTRACT: The MapReduce is an open source Hadoop framework implemented for processing and producing distributed large Terabyte data on large clusters. Its primary duty is to minimize the completion time of large sets of MapReduce jobs. Hadoop Cluster only has predefined fixed slot configuration for cluster lifetime. This fixed slot configuration may produce long completion time (Makespan) and low system resource utilization. The current open source Hadoop allows only static slot configuration, like fixed numbers of map slots and reduce slots throughout the cluster lifetime. Such static configuration may lead to long completion length as well as low system resource utilizations. Propose new schemes which use slot ratio between map and reduce tasks as a tunable knob for minimizing the completion length (i.e., makespan) of a given set. By leveraging the workload information of recently completed jobs, schemes dynamically allocates resources (or slots) to map and reduce tasks.. Many scheduling methodologies are discussed that aim to improve execution performance as well as completion time goal.

KEYWORDS: MapReduce, Makespan, Dynamic Job Ordering, Dynamic Slot Allocation.

I. INTRODUCTION

A cloud scheduler plays a main role in distributing resources for different jobs executing in cloud environment. Virtual machines are created and managed on the fly in cloud to create an environment for job execution. Map Reduce is a simple and powerful programming model which has been widely used for processing large scale data intensive applications on a cluster of physical machines. Now a day's many organizations, researchers, government agencies are running Map Reduce applications on public cloud. Running Map Reduce on cloud has many advantages like on-demand establishment of cluster, scalability. Many Map Reduce Frameworks like Google Map Reduce, Dryand, are available but the open source Hadoop Map Reduce is commonly used. But running a Hadoop cluster on a private cluster is different from running on public cloud .Public cloud enables to have virtual cluster where resources can be provisioned or released as per the requirement of the application in minutes. Executing Map Reduce applications on cloud enables user to execute jobs of different requirements without taking any pain of creating and maintaining a cluster. Scheduling plays a major role in the performance of Map Reduce Applications. The default scheduler in Hadoop Map Reduce is FIFO Scheduler, Face book uses Fair Scheduler, and Yahoo uses Capacity Scheduler. The above schedulers are typical examples of schedulers for Map Reduce application are best suitable for physical static clusters ,that can also serve the cloud systems with dynamic resource management, but these schedulers does not consider the features affected by virtualization used in cloud environments. Therefore, these is a need of dynamic scheduler which can schedule Map Reduce applications based on the features of the application, Virtual Machines and locality of input data to efficiently execute these applications in hybrid cloud environment.

II. REVIEW OF LETERATURE

MAPREDUCE SCHEDULING IN HADOOP

The most fundamental part of NameNode is job scheduling. FIFO (First In First Out) scheduling algorithm is the default scheduler used by HadoopMapReduce and designed for running large batch jobs. However users are given a chance to change their scheduling algorithm from FIFO to Fair scheduler or Capacity Scheduler. Fair scheduler is

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Website: www.ijirset.com

Vol. 6, Issue 7, July 2017

included with delay scheduler from the release of Hadoop 0.21. In FIFO scheduler, jobs are scheduled based on their job submission time and their priorities. This approach schedules one job to use all task slots and other jobs cannot use it until the current job completes. This increases the execution time of the jobs that are waiting ahead in the queue. Capacity Scheduler uses multiple queues/pools where each pool is guaranteed some fraction of physical resources in the cluster which enables more jobs to be executed concurrently. Fair scheduler creates pools for multiple users where each pool is guaranteed a share of resources fairly which results in fair share of the resources in the cluster and more jobs can be processed concurrently. The drawbacks of Capacity and Fair Scheduler are increased execution times and less sharing opportunities. Delay Scheduler of Hadoop does not impose strict rule of queuing for the tasks in the scheduling process.

In literature, there was research study on performance optimization of HadoopMapReduce jobs. An essential way for upgrading the performance of a MapReduce job by dynamic slot configuration and job scheduling.

J. Dean et al. 2008 [1] discussed MapReduce programming model. The MapReduce model performs operations using the map and reduces functions. Map function gets input from user documents. It generates intermediate key/value for reducing function. It further processes intermediate key/value pairs and provide output key/value pairs. At an entry level, MapReduce programming model provided the best data processing results. Currently, it needs to process the large volume of data. So it provides some consequences while processing and generating data sets. J. Polo et al. [2] calculated the map and reduce task completion time dynamically and update it every minute during job execution. Task scheduling policy was based on the priority of each job. Priority was estimated based on the concurrent allocation of jobs. J. Wolf et al. [3] implemented flexible scheduling allocation scheme with Hadoop fair scheduler. A primary concern is to optimize scheduling theory metrics, response time, makespan, stretch, and Service Level Agreement. They proposed penalty function for measurement of job completion time, epoch scheduling for partitioning time, moldable scheduling for job parallelization, and malleable scheduling for different interval parallelization.

Verma et al. [4] proposed deadline aware scheduler, called SLO scheduler. The SLO scheduler takes decisions of job ordering and slot allocation. This scheduler's primary duty is to maximize the utility function by implementing the Earliest Deadline First algorithms. J. Wang et al. [5] proposed fair slot setting for dynamically allocate available slots to particular tasks. They used FRESH for static and dynamic slot configuration. The static slot configuration slots are allocated before cluster launch based on previous task execution records. It uses deduct workload function to update current workloads of running jobs in the cluster. The fair scheduler was proposed to achieve fairness metric. The dynamic slot assignment slots are allocated during task execution.

S. Tang et al. [6] proposed three techniques to improve MapReduce performance. They categorized utilized slot into the busy slot and idle slot respectively. The primary concern is to increase the number of the busy slots and decrease number of idle slots. Dynamic Hadoop Slot Allocation observes idle map and reduce slots. DHSA allocated the task to the unallocated map slots for overflowed reduce slots. Speculative Execution Performance Balancing provides performance upgrade for a batch of jobs. Y. Yao et al. [7] proposed Tunable knob for reducing the Makespan of MapReduce (TUMM) for dynamic slot configuration. They modified the job tracker functionality by adding additional components. Main components are workload monitor and slot assigner. The workload monitor collects information about running and completed workloads. The slot assigner finds the best slot for dynamically assigning MapReduce tasks in the task tracker. They used FIFO schedulers for both static and dynamic slot configuration. They also introduced slot configuration for homogeneous and heterogeneous clusters. Wei and Tian [8] presented a genetic-algorithm-based (GA-based) scheduler for task level scheduling in HadoopMapReduce. Ge and Wei pointed out that the performance can further be improved by considering a realistic view of all the tasks that are waiting for processing. Guo & Tian [9] has proposed an optimal resource provisioning for Map reduce computing in public clouds. Ching-Chi Lin et. al in [10] presented an improved version of Round Robin algorithm used in Eucalyptus.

III. SYSTEM ARCHITECTURE

Design Goals

The design goals for Constraint Scheduler were:

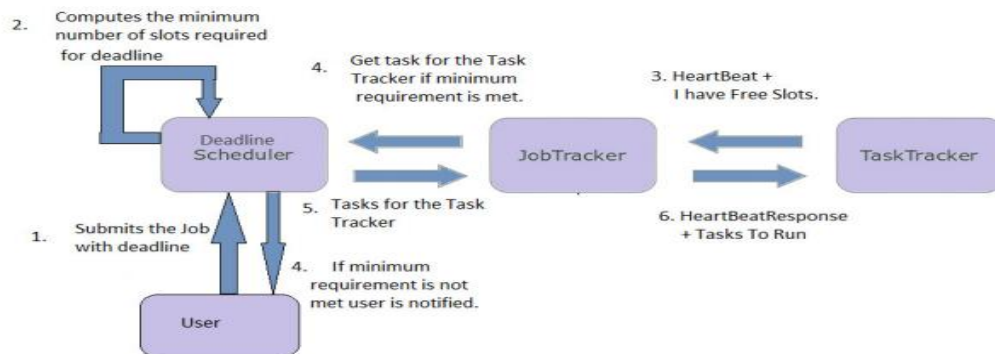
International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Website: www.ijirset.com

Vol. 6, Issue 7, July 2017

- 1) To be able to give users immediate feedback on whether the job can be completed within the given deadline or not and proceed with execution if deadline can be met. Otherwise, users have the option to resubmit with modified deadline requirements.
- 2) Maximize the number of jobs that can be run in the cluster while satisfying the time requirements of all jobs.



Fig, Block Diagram for System

The job submission process implemented by Job Submitter does the following:

Step 1: User submits the job.

Step 2: Scheduler will compute the minimum number of map and reduce slots required for the job completion.

Step 3: If schedulability test fails user will be notified to enter another deadline value.

Step 4: If passed, job will be scheduled.

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Website: www.ijirset.com

Vol. 6, Issue 7, July 2017

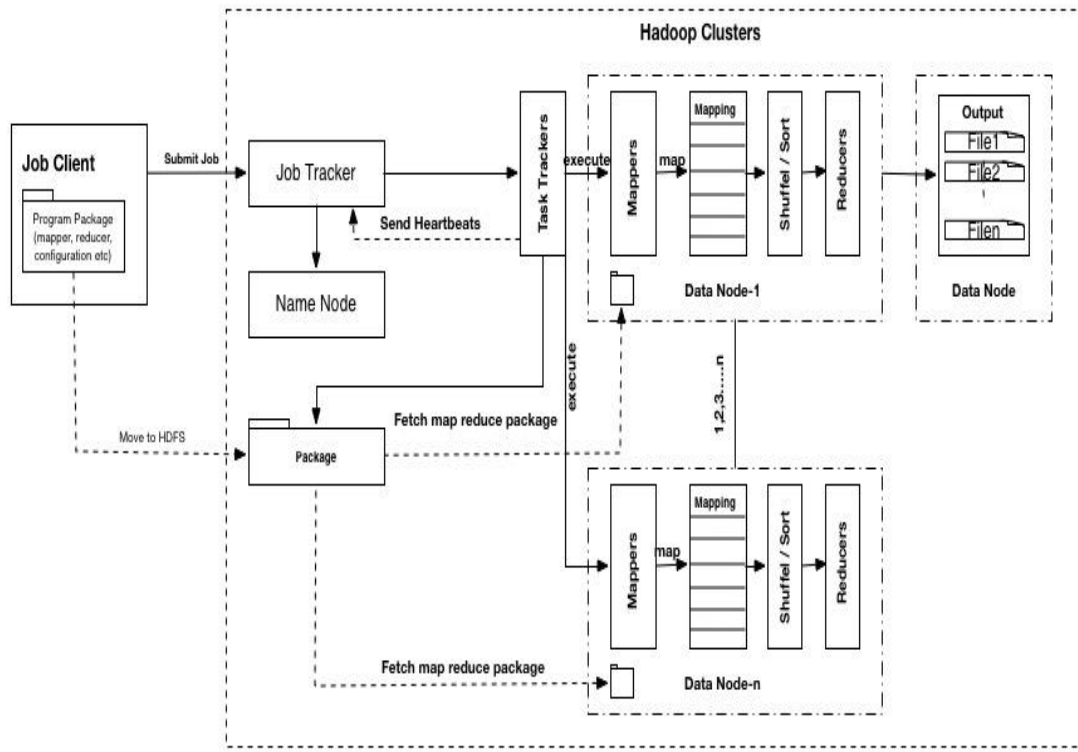


Fig. System Flow

IV. ALGORITHMS

Algorithm for Mapper:

Step 1: Map Runner reads a (key, value) pair from the assigned split in HDFS using Record Reader and applies the user-defined map function to the pair.

Step 2: After processing one pair, map task outputs the result to a fixed-size buffer. If the buffer is overflow it will apply a quick-sort to the full buffer first and then store the content of the buffer to a spill file so the buffer becomes empty and can therefore receive more results. This process will take place repeatedly until all records are used up.

Step 3: Map task merges all the spills on the disk into a sort file and stores it to the local file system as well as a corresponding index file referring to that data file. Further, task tracker will ask for a new task from job tracker as soon as the map task finishes and makes a slot free.

Algorithm for Reducer:

A reduce task execution comprises three phases.

Step 1: In the shuffle phase reduce task fetches a specific partition of every map task output using HTTP request and puts them into memory first. If the memory is full it will apply merge sort to the memory and output the result to a temp file.

Step 2: After all partition is fetched, reduce task enters the sort phase where it sorts all sorted files stored in memory or disk using merge sort grouping all records to the same key together.

Step 3: The reduce phase applies the user-defined reduce function to each key and the set of values belonging to the same key.

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Website: www.ijirset.com

Vol. 6, Issue 7, July 2017

Dynamic Job ordering Algorithm:

input : Estimating all Jobs' Map and Reduce durations (m_i, r_i) [1] by utilizing all available Map and Reduce slots in the system, the total number MapReduce slots ($P_{max\ m}$, $P_{max\ r}$) for a Hadoop cluster

output: scheduled jobs, makespan

Step 1: List the Map and Reduce's durations in two vertical columns (implemented in a list);

Step 2: for all jito J do

Step 3 Find the shortest one among all durations ($\min(m_i, r_i)$);

Step 4 In case of ties, for the sake of simplicity, order the item with the smallest subscript first. In case of a tie between Map and Reduce, order the item according to the Map;

Step 5 IF it is first Map type, place the corresponding item at the first place;

Step 6 ELSE it is first Reduce type, place the corresponding item at the last place;

Step 7 IF it is Map type, place the corresponding item right next to the previous job (i.e., in non-decreasing order);

Step 8 ELSE it is Reduce type, place the corresponding item left next to the previous job (i.e., in non-increasing order);

Step 9 Remove both time durations for that task;

Step 10 Repeat these steps on the remaining set of items;

Step 11 end

Step 12 Compute the makespan (C_{max});

V. RESULT

The system is built on hadoop 1.2.1 version and HDFS database. For dataset we use semi-structure xml data which consist of several attribute. Currently we compare windows system with Hadoop platform. The proposed system gives the estimated results. On the practical results shows the satisfactory level accuracy. Here following table shows the accuracy as well time complexity how it is better than traditional approaches.

Below table show that the uploading time and download time of txt file. In this we compare the performance of window system and hadoop system. Table show that window system is taking larger time for file upload and file download.

File Size	File Upload Time in ms	File Download Time in ms	File Upload Time in ms (By Hadoop)	File Download Time in ms (By Hadoop)
10(KB)	15	19	12	17
50(KB)	27.3	32.7	23.1	26.7
100(KB)	51.2	67.8	41	52.6
1(MB)	250.8	200	220.5	180
3(MB)	400.7	350.4	350.9	320
5(MB)	550	480.9	450	420.4

Table: File Upload Time and Download Time on Window System and Hadoop System

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Website: www.ijirset.com

Vol. 6, Issue 7, July 2017

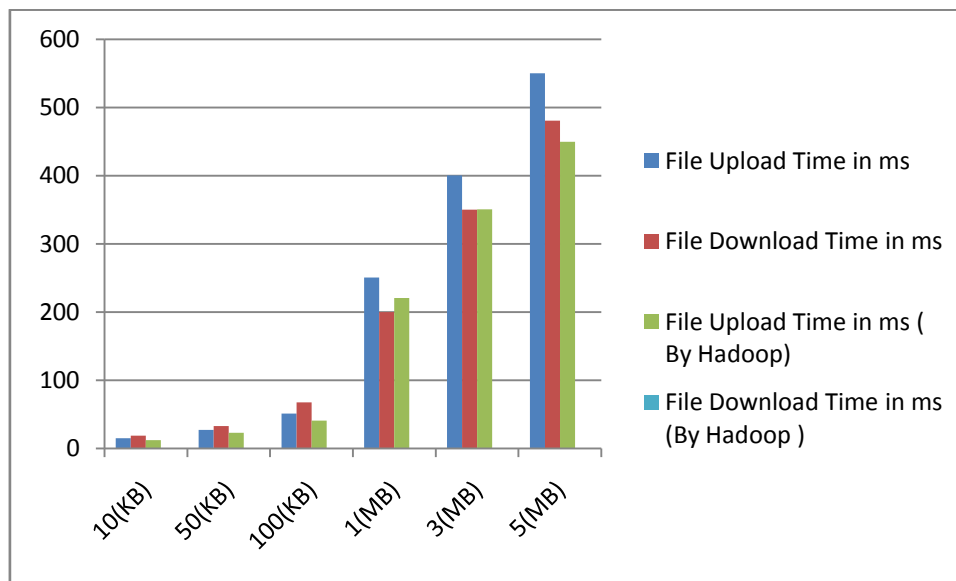


Fig.System Performance Measure Window vsHadoop.

VI. CONCLUSION

In current day world as there is huge increase in volumes of data and big data has become an important point of research. This paper has discussed scheduling of Map Reduce parallel applications on cloud. There have been an active research in the area of scheduling of the map and reduce tasks to virtual machines to improve the performance of map reduce applications. Most of the scheduling algorithms concentrate on map tasks data locality. Scheduling can be made efficient by using the knowledge of data locality of the intermediate data generated by the map tasks. This knowledge helps out to reduce the intermediate network traffic during the reduce phase and there by speeding the execution of map reduce applications.

We extended the approach of real time cluster scheduling to derive minimum map and reduce task count criteria for performing task scheduling with deadline constraints in Hadoop. We computed the amount of resource required to complete a job for a particular deadline.

REFERENCES

- [1] Jeffrey Dean, Sanjay Ghemawat. MapReduce: Simplified Data Processing on Large Clusters Operating Systems Design and Implementation.
- [2] J. Polo, D. Carrera, Y. Becerra, J. Torres, E. Ayguad'e, M. Steinder, and I. Whalley. Performance-driven task co-scheduling for MapReduce environments. In 12th IEEE/IFIP Network Operations and Management Symposium.ACM, 2010.
- [3] Joel Wolf, Deepak Rajan, Kirsten Hildrum, Rohit Khandekar, Kun-Lung Wu. FLEX: A Slot Allocation Scheduling Optimizer for MapReduce Workloads, In IFIP International Federation for Information Processing 2010.
- [4] Abhishek Verma, Ludmila Cherkasova, Roy H. Campbell. ARIA: Automatic Resource Inference and Allocation for MapReduce Environments. In Proceedings of the 8th IEEE International Conference on Autonomic Computing (ICAC'2011), 2011.
- [5] Jiayin Wang, Ying Mao. FRESH: Fair and Efficient Slot Configuration and Scheduling for Hadoop Clusters. DOI:10.1109/CLOUD.2014.106
- [6] Shanjia Tang, Bu-Sung Lee, Bingsheng He. DynamicMR: A Dynamic Slot Allocation Optimization Framework for MapReduce Clusters. In IEEE TRANSACTIONS ON CLOUD COMPUTING.
- [7] Yi Yao, Jiayin Wang, Bo Sheng, Chiu C. Tan. Self-Adjusting Slot Configurations for Homogeneous and Heterogeneous Hadoop Clusters. In IEEE Transactions on Cloud Computing, DOI 10.1109/TCC.2015.2415802.
- [8] Yibin Wei, Ling Tian, Research on cloud design resources scheduling based on Genetic Algorithm, 2012 International Conference on systems and informatics (ICSAI 2012)



ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Website: www.ijirset.com

Vol. 6, Issue 7, July 2017

- [9] Keke Chen, James Powers, ShuminGuo, and FengguangTian. CRESP: Towards Optimal Resource Provisioning for MapReduce Computing in Public Clouds. In IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 25, NO. 6, JUNE 2014.
- [10] Ching-Chi Lin, Pangfeng Liu, and Jan-JanWu. Energy-aware virtual machine dynamic provision and scheduling for cloud,. In Cloud Computing (CLOUD), 2011 IEEE International Conference on, pages 736 –737, july 2011.