ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering
*(An ISO 3297: 2007 Certified Organization)*
*Website: www.ijircce.com*
**Vol. 5, Issue 2, February 2017**

# Dynamic MapReduce for Job Workloads through Slot Configuration Technique

K. Venu Gopal[1], I. Kavitha Jackleen[2]

[1]M.Tech (CSE)., Dept of CSE, Global College of Engineering and Technology, Kadapa, Andhra Pradesh, India

[2]Assistant Professor, Dept of CSE, Global College of Engineering and Technology, Kadapa, Andhra Pradesh, 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:** Map Reduce, Makespan, Workload, Dynamic Slot Allocation.

## I.NTRODUCTION

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, Facebook 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.MOTIVATION

In this part discuss the important characteristics of my proposed algorithm, based on the challenges of the Hadoop system.

**1. Scheduling based on fairness, minimum share requirements, and the heterogeneity of jobs and resources.**
In a Hadoop system satisfying the minimum shares of the users is the first critical issue. The next important issue is fairness. I design a scheduling algorithm which has two stages. In the first stage, the algorithm considers the satisfaction of the minimum share requirements of all the users. Then, in the second stage, the algorithm considers fairness among all the users in the system. Most current Hadoop scheduling algorithms consider fairness and minimum

share objectives without considering the heterogeneity of the jobs and the resources. One of the advantages of my proposed algorithm is that while our proposed algorithm satisfies the fairness and the minimum share requirements, it

further matches jobs with resources based on job features (e.g. estimated execution time) and resource features (e.g. execution rate). Consequently, the algorithm reduces the completion time of jobs in the system.

**2. Reducing communication costs.**

In a Hadoop cluster, the network links among the resources have varying bandwidth capabilities. Moreover, in a large cluster, the resources are often located far from each other. The Hadoop system distributes tasks among the resources to reduce a job's completion time. However, Hadoop does not consider the communication costs among the resources. In a large cluster with heterogenous resources, maximizing a task's distribution may result in significant communication costs. Therefore, the corresponding job's completion time will be increased. In our proposed algorithm, we consider the heterogeneity and distribution of resources in the task assignment.

## III.LITERATURE SURVEY

In literature, there was research study on performance optimization of Hadoop MapReduce jobs. An essential way for upgrading the performance of a MapReduce job is 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. It takes much execution time for task initialization, task coordination, and task scheduling. Parallel data processing may lead to inefficient task execution and low resource utilization.

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. The dynamic scheduler is pre-emptive. It affects resource allocation of low priority 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. [5] 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. It measures how many numbers of slots required for scheduling the slots dynamically with a particular job deadline.

B. Sharma et al. [7] proposed a global resource manager for the job tracker and a local resource manager for the task tracker. A global resource manager function is to manage each MapReduce task. It processes resource needs and resource assignments for each task. A local resource manager's duty is to identify each task. It examines resource usage and task completion time of the task. It deals with detecting bottlenecks with resources and resource contention.

Apache Hadoop released next generation MapReduce, called YARN [8]. It replaces MRv1 fixed slot configuration. YARN deals with CPU cores and memory requirements. It splits the job tracker into two components; they are resource managements and job scheduling. MapReduce tasks assignment is based on CPU cores and memory requirement of each task. YARN users simply update their MRv1 by installing mrv2 compatibility API and recompile the MRv1 application.

J. Wang et al. [9] 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. It used Johnson indices to represent the level of fairness.

S. Tang et al. [11] 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

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering
*(An ISO 3297: 2007 Certified Organization)*
*Website: www.ijircce.com*
**Vol. 5, Issue 2, February 2017**

performance upgrade for a batch of jobs. It gives the highest priority to failed tasks and next level priority to pending tasks. The slot prescheduling improves the performance of slot utilization with data locality without any negative

effects on fairness metric.
A.U. Patil et al. [13] discussed scheduling algorithms in the MapReduce environment. They analyzed schedulers and scheduling policies. The default FIFO scheduler follows the First in First Out queue for schedules the job. A single job is divided into a small number of chunks called tasks. The FIFO queue allocates tasks to free slots presented in the task tracker. The fair scheduler provides the fair share of resources to cluster users. Capacity scheduler estimates the number of users sharing cluster resources and focus fair allocation of resources to users. The primary concern is to maximize the throughput and utilization of entire cluster. Scheduling policies include the Longest Approximation Time to End, Deadline constraint, delay scheduling, resource aware, and Fair4s scheduling.
Z. Liu focused [14] partition skew problem. Data skewness causes the problem in execution time for larger and smaller tasks. Commonly this can be raised while partitions are unevenly distributed by the hash function. They proposed a new architecture called DREAMS. It predicts partition size and estimates reduce task performance metrics like CPU and memory impacts. Reduce phase performance model also detects the relationship between partition size and task execution time. After completion of reduce task performance estimation, DREAMS allocates resources to tasks.
Y. Yao et al. [15] 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. For the heterogeneous environment, H_TUMM slot assignment algorithm was implemented. Authors used work count, histogram rating, classification, inverted index, and grep jobs for experimental results.

## IV.SYSTEM ARCHITECTURE

### Design

### Goals

The design goals for Constraint Scheduler were:
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.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering
(An ISO 3297: 2007 Certified Organization)
Website: www.ijircce.com
Vol. 5, Issue 2, February 2017

**System Flow**

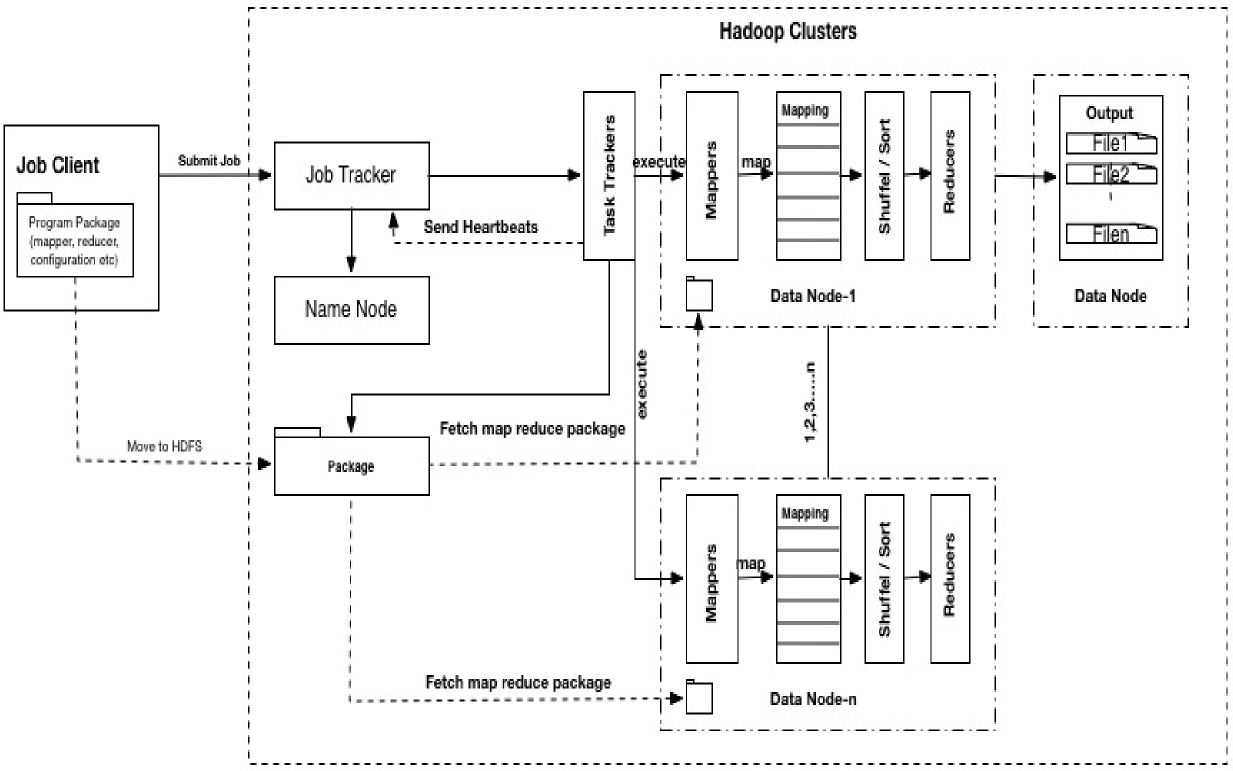

**Fig: System Flow**

## V.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. For this, we proposed the idea of estimation of the values of parameters: filter ratio, cost of processing a unit data in map task, cost of processing a unit data in reduce task, communication cost of transferring unit data. Our observation shows that for a particular data size with increasing deadlines, resource demand will decrease. Also, if the data size increases and deadline is kept constant, resource demand will increase. If resource demand increases, we can meet the demand by adding physical or virtual node to the existing cluster dynamically or provide a feasible deadline. We have implemented the same.
For the future enhancement we can focus on Dynamic slot configuration, it is one of the important factors while processing a large data set with MapReduce paradigm. It optimizes the performance of MapReduce framework. Each job can be scheduled using any one of the scheduling policies by the job tracker.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

**International Journal of Innovative Research in Computer and Communication Engineering**

*(An ISO 3297: 2007 Certified Organization)*

*Website: www.ijircce.com*

**Vol. 5, Issue 2, February 2017**

## REFERRENCES

[1]http://hadoop.apache.org/docs/r1.2.1/fairscheduler.html
[2]http://hadoop.apache.org/docs/r2.3.0/hadoop-yarn/hadoop yarnsite/CapacityScheduler.html
[3] 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.
[4] Anton Beloglazov and Rajkumar Buyya. Energy efficient allocation of virtual machines in cloud data centers, In 10th IEEE/ ACM International Conference on Cluster, Cloud and Grid Computing, pages 577–578, 2010.
[5] Yibin Wei, Ling Tian , Research on cloud design resources scheduling based on Genetic Algorithm, 2012 International Conference on systems and informatics(ICSAI 2012)
[6] Chen, K. ; Powers, J. ; Guo, S. ; Tian, F. CRESP: Towards Optimal Resource Provisioning for MapReduce Computing in Public Clouds , IEEE Transactions on Parallel and Distributed Systems ,Volume: 25 , Issue: 6 Publication Year: 2014 , Page(s): 1403 – 1412.
[7] Xiaohong Zhang ; Yuhong Feng ; Shengzhong Feng ;Jianping Fan ; Zhong Ming An effective data locality aware task scheduling method for MapReduce framework in heterogeneous environments, 2011 International Conference on Cloud and Service Computing (CSC)Year: 2011 , Page(s): 235 - 242 [8] Sewoog Kim ; Dongwoo Kang ; Jongmoo Choi ; Junmo Kim Burstiness-aware I/O scheduler for MapReduce framework on virtualized environments , 2014 International Conference on Big Data and Smart Computing (BIGCOMP) Publication Year: 2014 , Page(s): 305 – 308.
[9] Hammoud, M. ; Rehman, M.S. ; Sakr, M.F. Center-ofGravity Reduce Task Scheduling to Lower MapReduce Network Traffic , 2012 IEEE 5th International Conference on Cloud Computing (CLOUD) Publication Year: 2012 , Page(s): 49 – 58.
[10] 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.
[11] L. Phan, Z. Zhang, B. Loo, and I. Lee. Real-time MapReduce Scheduling. Tech. Report No. MS-CIS-10-32, UPenn, 2010.
[12] B. Palanisamy, A. Singh, L. Liu, and B. Jain. Purlieus: localityaware resource allocation for MapReduce in a cloud. In Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC), 2011.
[13] Resource management with VMware DRS http://www.vmware.com/pdf/vmware_drs_wp.pdf.