THOMSON REUTERS ENDNOTE

[Ramla, 5(1): January,2016] ISSN: 2277-9655
(I2OR), Publication Impact Factor: 3.785

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

SIGNIFICANCE OF VARIOUS HADOOP JOB SCHEDULERS-A RETROSPECTIVE

M.Ramla,MCA M.Phil., M.Ramesh, M.Sc., M.Phil.
Assistant Professor, Department of Computer Science, SRM University, India

ABSTRACT

We live in the era of data explosion and Data Analysis is an important functionality as deluge of data comes from distinctive domain. The five V's of data-Volume, Velocity, Variety, Veracity, Value brings in a robust computation model known as Hadoop, which is in huge demand in market nowadays. The centerpiece of Hadoop is the Schedulers. A hadoop scheduler quickly multiplex the incoming jobs on available resources. To boost the performance of Hadoop schedulers is very essential. With this motivation, this paper delves into the various job scheduling algorithms in Hadoop. Comparative study of Hadoop scheduler from varied parametric aspects is the essence of this paper.

KEYWORDS: Hadoop, MapReduce, HDFS, Schedulers

INTRODUCTION

Nowadays dealing with sheer volume of datasets in the order of Yottabytes and Zetabytes is a reality[1]. The existing storage capacity is not enough to leverage the power of massive parallel processing. HADOOP - A well adopted, standards-based, open source software framework has rapidly become the industry and academic standard. Hadoop is the software framework for processing large data sets. The advantage of Hadoop is that you can combine data storage and processing [8]. HDFS used for storage and MR for processing are the two components of Hadoop [3]. A scheduler which plays a crucial role in the performance of big data processing is our primary concern.

The paper is structured as follows: In section I, brief summary of Hadoop system is given. Section II deals with the various scheduling algorithms and the pros and cons is presented. Finally conclusion remarks and the future work are proposed.

HADOOP OVERVIEW

Hadoop is Java based programming model for large data set processing in distributed environment sponsored by Apache Software foundation [8]. It provides much needed robustness and scalability options to a distributed system. It is fault tolerant and can be deployed on low cost hardware[4]. The storage system is not physically separated from processing system. MR is the programming model for processing large data sets and HDFS is used to stream those large data sets.

HDFS:

Hadoop Distributed File System is Hadoop's implementation designed to hold a large amount of data and provide access to data to many clients across network. It comprises of two nodes- Data node for storing data and ame node (master node) for monitoring data nodes[8].Scheduling decisions are taken by Master nodes called Job trackers and the slave nodes called Task Trackers execute the tasks[2]. HDFS is resilence, fault tolerant and it also minimizes disk seeks.

ISSN: 2277-9655
(I2OR), Publication Impact Factor: 3.785

Map-Reduce MR

A programming representation for processing sizable datasets. MR lets you crunch massive amounts of information. Data is put as key-value pairs. There are two types of slot known as Map slot and Reduce slot. Each map or reduce task finishes within 30-40 seconds[9].

Map Algorithm includes 3 steps[11]

(i) Provide a map task for each input split.
(ii) Execute Map task
(iii) Mappers output is stored and allocted to each reducer.

Reduce Algorithm includes 3 steps [11]

(i) Assign related block for each reducer (Shuffle)
(ii) Input is grouped according to the key
(iii) Secondary sorting is done.

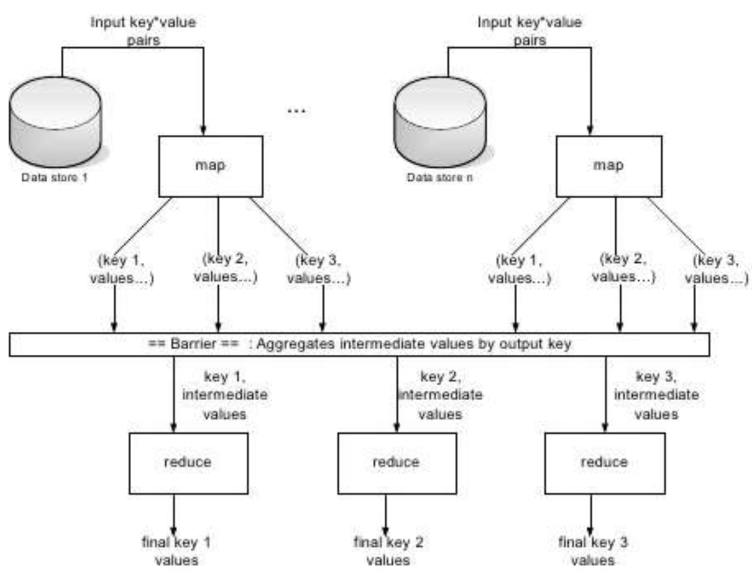

Figure 1: A Map Reduce Computation

EARLY SCHEDULERS

FIFO

This is the default scheduler of Hadoop. In this scheduling, a Job tracker pulls the job from the queue , oldest job first as they get acquirable free resources[2]. This treats a jobs importance relative to when it was submitted. It is simple and efficient.

Fair Share

The core idea is to equally distribute computing resources among users or jobs in the system over time. Hadoop creates a set of pools into which the jobs are placed for selection by scheduler. Each pool has equal shares to resources. To ensure fairness, each user is assigned to a pool. When only one application is running, the entire cluster will be used by the application. But when other applications are submitted, resources that gets free are assigned to the new applications, so that each application eventually gets approximately the same amount of resources. Small jobs intermix with longer jobs and finish quickly. Thus user heterogeneity is considered in this scheduler. The Quality of Service (QoS) is also improved [5].

Capacity Scheduler

This scheduler is similar to that of fair share with distinct differences. It maximizes the throughput and utilization of cluster. It provides elasticity for organizations in a cost-effective manner. In capacity scheduling, instead of pools, several queues are created, each with a configurable number of map and reduce slots. Each queue is also assigned a guaranteed capacity of resources[3]. Queues are monitored; if a queue is not consuming its allocated capacity, this excess capacity can be temporarily allocated to other queues.

IMPROVEMENTS IN SCHEDULERS

LATE – Longest Approximate Time to End

The goal of this scheduler is to minimize the response time. The straggler tasks are identified based on the progress score and it is run as a clone on fast nodes[3]. A threshold is defined for selecting the speculative execution. Sometimes running speculative tasks on some jobs may degrade performance. But launching a few extra speculative tasks is not harmful.

Delay Scheduling

The delay scheduling relaxes the queuing policy for limited time to achieve locality. If the head of line job cannot start locally then skip and look for subsequent jobs. [8] After a threshold value is reached, the skipped job is allowed to start non-local execution to prevent starvation. Very short time (1-5s) is enough to get nearly 100% locality.

MAESTRO

This scheduler is a replica aware scheduler. It does the work in two waves (i) Fills the empty slots of each data node based on the number of hosted map task and on the replication scheme for their input data. (ii) Runtime scheduling takes into account the probability of scheduling a map task on a given machine depending on the replicas of the task's input data. [8] These two waves lead to higher locality in the execution of map tasks.

CREST - Combination Re execution Scheduling tasks

This scheduler is better than LATE and brings improvement by re-executing a combination of tasks on a group of nodes[8]. CREST can achieve optimal running time for speculative map tasks. The main idea is that re-executing a combination of tasks on a set of cluster nodes may improve the scheduler performance than directly speculating the straggler task on a target node, due to data locality.

LARTS-Location Aware Reduce Task Scheduler

LARTS uses a practical strategy that leverages network locations and sizes of partitions to exploit data locality. This is specifically for reduce tasks. Schedule the reducers as close as possible to their maximum amount of input data [8]. Ultimately network traffic is reduced. Awareness of partitions, locations and size is required for scheduling. The data access delay that degrades the system performance is reduced.

Context-Aware Scheduling

This scheduler takes into account the job characteristics and the available resources within cluster nodes. The three steps used to perform its objective are:

(i) Classify the jobs as CPU bound or IO bound.
(ii) Classify nodes as computational or good
(iii) Map the task to nodes based on demand.[8]

Center of Gravity Reduce Scheduler (CoGRS)

This scheduler attempts to schedule every reduce task at its center of gravity node determined by its network location [8]. This decreases the Network traffic and allow more MR jobs to reside on the same system.

COSSH- Classification and Optimization Scheduling for Heterogeneous Hadoop

COSSH considers heterogeneity in both application and cluster level. This scheduler uses system information to make better scheduling decisions. It receives a new job and places it in appropriate queue. When heart beat is received, assign a job to the current free resource. The mean completion time of jobs is considerably improved[8]

Resource-Aware Scheduler

Each task tracker monitors the resources such as CPU utilization, Disk I/O, number of page faults etc[3]. This reflects the actual processing power of nodes. RAS determines the number of job slots, and their cluster, lively at run-time. This contrasts sharply with the traditional approach of requiring the system administrator to statically and homogeneously configure the slot count and type on a cluster. This eases the configuration burden and improves the behavior of MR cluster.

Deadline Aware Scheduler

This scheduler addresses the issue of deadlines. Production jobs varies significantly in very many aspects like urgency, utility and structure[6]. Current schedulers typically do not support hard/soft deadlines. So, deadline awareness is required for jobs as they are used for business critical decisions of data. This will significantly improve the productivity of the business.

Table 1: Hadoop Schedulers with its Pros & Cons

Scheduling Algorithm	***Feature***	***Pros***	***Cons***
FIFO	First Come First Serve	1. High Throughput 2. Production jobs completes on time	1. Performance degrades for small jobs 2. Low utilization of resources 3. No heterogeneity of workload and performance constraints is considered
Fair Share	User heterogeneity	1. Suited for both small and large clusters 2. Short jobs finish in reasonable time intermixed with longer jobs 3. Greater responsiveness of cluster.	1. Larger average completion time 2. Job weight of each node is not considered
Capacity	User and Job heterogeneity with Fairness	1. High utilization of resources 2. Supports Preemption 3. Faster Response time 4. Cluster Stability is improved 5. Elasticity, Security, multi-tenancy operability, fairness	1. User needs to know system information and make queue set for the job. 2. Configuration Complicated.
LATE	Speculative Tasks	1. Robust, improves overall job performance	1. No sync between mappers and reducers. 2. Not always reliable due to bugs in tasks
Delay	Node-aware	1. Data locality is considered 2. Efficient as tasks are run near their input data. 3. Fairness is maintained 4. Sticky slots resolved 5. Head-of-line scheduling resolved	Relaxes fairness slightly
MAESTRO	Replica-aware	1. Provide high locality in execution of map tasks 2. Balanced data distribution for shuffling phase	-
CREST	Combined speculative execution of tasks	1. Response time is improved 2. Better than LATE, optimal running time for speculating map task	Re execution may degrade the performance at times
LARTS	Location Aware Reduce tasks	1. Network traffic is reduced 2. Balances scheduling delay, skew, system utilization and parallelism	Static sweet spot determination

Context Aware	CPU, Network, disk requirements characteristics	1. Heterogenity of cluster and workload mix is considered	Still in simulation stage
CoG	Locality & Skew Aware	1. Decresed network traffic 2. More MR jobs consists on the same system	Complex
COSSH	Classify resources and optimize the performance	1. Improves mean completion time of jobs 2. Cluster, workload, user heterogeneity is considered	Search overhead
Resource Aware	Actual Power of the task trackers	1. CPU Utilization, disk channel IO, number of page faults, VM state, disk channel loading are considered	Each Task Tracker [TT] node to monitor its resources.
Deadline Aware	Urgency, utility and structure of Jobs considered	1. Crucial decision making can be done on time 2. Improves the productivity of business	Some jobs that are not under priority is likely to suffer

CONCLUSION

Hadoop is addressing the Big Data Challenges and coping up with the trend in data explosion. We embark on understanding the various Hadoop Schedulers and it our humble expectation that the paper serves as a first stop for beginners to get an insight into the various job schedulers in Hadoop.In this paper, we have analyzed various schedulers from very many aspects like fairness, synchronization, locality-aware, speculative execution of tasks, resource-aware, context-aware, deadline-aware. However for an efficient processing of MR applications that runs in data centers, energy costs is a crucial factor. Efficient spatial placement of tasks on nodes will maximize the utilization and provide energy savings in data centers. Incorporating energy aware scheduling for multiple MR jobs will be our extended work.

REFERENCES

1. Jens Dittrich, Jorge-Arnulfo Quiane-Ruiz, "'Efficient Big Data Processing in Hadoop MR", The 38th international conference on very large database, Vol. 5, No. 12.
2. A.U.Patil, T.I.Bagban, A.P.Pande, "Recent Job Scheduling Algorithms in Hadoop Cluster Environments, A Survey", International journal of Advanced Research in computer and communication Engineering, Vol. 4, Issue 2, Feb. 2015.
3. Harshavardhan S.Bhosale, Devendra P. Gadekar,"Big Data Processing using Hadoop : Survey on Scheduling", Intrenaltional journal of Science and Research, Vol. 3, Issue 10, Oct. 2014.
4. Suman Arora, Dr. Madhu Goel, "Survey paper on Scheduling in Hadoop", IJARCSSE, Vol. 4, Issue 5, May 2014.
5. Jilan Chen, Dan Wang and Wenbing Zhao, "A task scheduling Algorithm for Hadoop Platform", Journal of Computers Vol. 8, No. 4, April 2013.
6. Peter Bodik, Ishai Menache, Joseph Naor, "Brief Announcement: Dead-line Aware scheduling of Big Data processing jobs", SPAA'14, June 2014.
7. Lena Mashayekhy et al., "Energy-Aware Scheduling of Map Reduce Jobs for Big Data Aplications", IEEE transactions on parallel and distributed Systems, Vol. 25, 2014.
8. Seyed Reza Pakize, "A Comprehensive view of Hadoop MR Scheduling Algorithms", Internaltional journal of computer networks and communications security, Vol. 2, No. 9, Sep 2014.

ISSN: 2277-9655
(I2OR), Publication Impact Factor: 3.785

9. Sasiniveda.G, Revathi.N,"Performance Tuning and scheduling of Large data set analysis in Map Reduce Paradigm by Optimal Configuration using Hadoop", International Journal of Computer Applications, Vol.70 No.21, May 2013

AUTHOR BIBLIOGRAPHY

	**M.RAMLA** Received MCA, M.Phil Degree in Computer Science from Manonmaniam Sundaranar University, Tamil Nadu, India. She is currently working as Assistant Professor in SRM University. Her research interests include Databases, Distrbuted Systems, Cloud Computing.
	**M.RAMESH** Received M.Sc, M.Phil Degree in Computer Science from Madurai Kamaraj University, Tamil Nadu, India. He is currently working as Assistant Professor in SRM University. His research interests include Database Management, Distrbuted Systems.