ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE) Vol. 3, Special Issue 7, March 2017

Hadoop Performance Modelling for Job Estimation and Resource Allocation

S.PONMANI [1], Dr.C.SUMITHRADEVI [2]

1. P.G. Student, Dept. of MCA, VSB Engineering College, Karur, Tamilnadu, India
2. Assistant Professor, Dept. of MCA, VSB Engineering College, Karur, Tamilnadu, India

ABSTRACT: Map Reduce has become a major computing model for data intensive applications. Hadoop, an open source implementation of Map Reduce, has been adopted by an increasingly growing user community. Cloud computing service providers such as Amazon EC2 Cloud offer the opportunities for Hadoop users to lease a certain amount of resources and pay for their use. However, a key challenge is that cloud service providers do not have a resource provisioning mechanism to satisfy user jobs with deadline requirements. Currently, it is solely the user's responsibility to estimate the required amount of resources for running a job in the cloud. This work presents a Hadoop job performance model that accurately estimates job completion time and further provisions the required amount of resources for a job to be completed within a deadline. The proposed model builds on historical job execution records and employs Locally Weighted Linear Regression (LWLR) technique to estimate the execution time of a job. Furthermore, it employs Lagrange Multipliers technique for resource provisioning to satisfy jobs with deadline requirements.

1. INTRODUCTION

Many organizations are continuously collecting massive amounts of datasets from various sources such as the World Wide Web, sensor networks and social networks. The ability to perform scalable and timely analytics on these unstructured datasets is a high priority task for many enterprises. It has become difficult for traditional network storage and database systems to process these continuously growing datasets. MapReduce, originally developed by Google, has become a major computing model in support of data intensive applications. It is a highly scalable, fault-tolerant and data parallel model that automatically distributes the data and parallelizes the computation across a cluster of computers. Among its implementations such as Mars, Phoenix, Dryad and Hadoop, Hadoop has received a wide uptake by the community due to its open source nature. Building on the HP model, this system presents an improved HP model for Hadoop job execution estimation and resource provisioning. The major objectives of the system are as follows:

- The improved HP work mathematically models all the three core phases of a Hadoop job. In contrast, the HP work does not mathematically model the non-overlapping shuffle phase in the first wave.
- The improved HP model employs locally weighted linear regression (LWLR) technique to estimate the execution time of a Hadoop job with a varied number of reduce tasks. In contrast, the HP model employs a simple linear regress technique for job execution estimation which restricts to a constant number of reduce tasks.

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering
(IJARMATE) Vol. 3, Special Issue 7, March 2017

- Based on job execution estimation, the improved HP model employs Lagrange Multiplier technique to provision the amount of resources for a Hadoop job to complete within a given deadline.

2. LITERATURE SURVEY

A. DYNAMIC SLOT ALLOCATION TECHNIQUE

Dynamic Hadoop Fair Scheduler(DHFS)

It schedules the job in order manner, it has pool-independent DHFS(PI-DHFS) and pool-dependent DHFS(PD-DHFS).Pool dependent DHFS have each pool which satisfy only its own map and reduce tasks with its shared map and reduce slots between its map-phased pool and reduce-phased pool,it is known to be intra pool dynamic slots allocation. Pool independent DHFS considers the dynamic slots allocation from the cluster level itself, instead of pool-level. The map tasks have priority in the use of map slots and reduce tasks have priority to reduce slots are called as intra phase dynamic slots allocation. When the respective phase slots requirements met excess slots be used by other phase are referred as inter phase dynamic slots allocation.

Dynamic slot allocation

It is a technique for managing global session resources and it allows dynamic resizing of the cache per-client, per-load basis. The client communicates to the server about whether resources are filled in all slots or not filled in the slots. The server then decides how many slots it should allocate to that client in the future. Communication occurs via the sequence operation, which means that updates occur on every step.

Process of Hadoop Fair Scheduler

It runs small jobs quickly, even if they are sharing a cluster with large jobs. The users should only need to configure, it support reconfigure at runtime without requiring a cluster restart. Fair scheduling is a method of assigning resources to jobs. When there is a single job running, that job uses the entire cluster. When other jobs are submitted, tasks slots that free up are assigned to the new jobs, so that each job gets roughly the same amount of CPU time.

B. JOB SCHEDULING FOR MULTI-USER MAPREDUCE CLUSTERS

Job scheduling process

Job scheduling in Hadoop is performed by the job master,which manages a number of worker nodes in the cluster. Each worker has a fixed number of map and reduce slots,which can run tasks.The workers periodically send heartbeats to the master to reporting the number of free slots and tasks. The objective of scheduling is to determine the job schedules that minimize or maximize a measure of performance. It can be characterized by a set of jobs, each with one or more operations.

Multi user cluster scheduling

The jobs are composed of small independent tasks,it is possible to isolate while using cluster

efficiently. The steps in multi user cluster scheduling are to make tasks small in length having small tasks make other new jobs to startup quickly.

Delay scheduling

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering
(IJARMATE) Vol. 3, Special Issue 7, March 2017

The job which is to be scheduled next, should wait for the previous job to be completed. After completion of the previous job,the current job should be processed and delay occurs. This type of job scheduling are known as delay scheduling.

Copy compute splitting

HDFS copy tasks that because they perform large amounts of memory copies when merging map outputs, there is little gain from copy-compute (copy tasks compete with compute tasks with CPU).

2.1 EXISTING SYSTEM

Meeting job deadlines is difficult in current Hadoop platforms. First, because jobs have diverse resource demands, it is hard to determine how much resource is needed for each job to avoid its deadline miss. Second, Hadoop clusters are usually shared by multiple jobs and the scheduling order of these jobs can affect job completion time]. Thus, allocating sufficient resources alone may not guarantee job completion time effectively. While existing schedulers in Hadoop, such as the default FIFO scheduler, Fair scheduler, Capacity scheduler, the RAS scheduler, and their variations, optimize job completion time without considering deadlines.

2.1.1 DISADVANTAGES

- Current use of Hadoop in research and enterprises still has significant room for improvement on the performance of Hadoop jobs and the utilization of Hadoop clusters. There is a growing need for providing predictable services to Hadoop users who have strict requirements on job completion times (i.e., deadlines).
- Does not consider the mapping between VM's and hosts. I.e no load balancing is done.
- Longer jobs always tend to get pushed back, as shorter jobs get priority.
- It uses static allocation. Does not consider dynamic allocation

2.2 PROPOSED SYSTEM

MapReduce has become a major computing model for data intensive applications. Hadoop, an open source implementation of MapReduce, has been adopted by an increasingly growing user community. Cloud computing service providers such as Amazon EC2 Cloud offer the opportunities for Hadoop users to lease a certain amount of resources and pay for their use. However, a key challenge is that cloud service providers do not have a resource provisioning mechanism to satisfy user jobs with deadline requirements. Currently, it is solely the user's responsibility to estimate the required amount of resources for running a job in the cloud. The proposed system presents a Hadoop job performance model that accurately estimates job completion time and further provisions the required amount of resources for a job to be completed within a deadline. The proposed model builds on historical job execution records and employs Locally Weighted Linear Regression (LWLR) technique to estimate the execution time of a job. Furthermore, it employs Lagrange Multipliers technique for resource provisioning to satisfy jobs with deadline requirements.

2.2.1 BENFITS OF PROPOSED SYSTEM

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering
(IJARMATE) Vol. 3, Special Issue 7, March 2017

- The improved HP work mathematically models all the three core phases of a Hadoop job. In contrast, the HP work does not mathematically model the non-overlapping shuffle phase in the first wave.
- The improved HP model employs Locally Weighted Linear Regression (LWLR) technique to estimate the execution time of a Hadoop job with a varied number of reduce tasks. In contrast, the HP model employs a simple linear regress technique for job execution estimation which restricts to a constant number of reduce tasks.
- Based on job execution estimation, the improved HP model employs Langrage Multiplier technique to provision the amount of resources for a Hadoop job to complete within a given deadline.

3. SYSTEM DESCRIPTION

Recently, a number of sophisticated Hadoop performance models are proposed. Starfish collects a running Hadoop job profile at a fine granularity with detailed information for job estimation and optimization. On the top of Starfish, Elasticiser is proposed for resource provisioning in terms of virtual machines. However, collecting the detailed execution profile of a Hadoop job incurs a high overhead which leads to an overestimatedjob execution time. The HP model considers both the overlapping and non-overlapping stages and uses simple linear regression for job estimation. This model also estimates the amount of resources for jobs with deadline requirements. CRESP estimates job execution and supports resource provisioning in terms of map and reduce slots. However, both the HP model and CRESP ignore the impact of the number of reduce tasks on job performance. The HP model is restricted to a constant number of reduce tasks, whereas CRESP only considers a single wave of the reduce phase. In CRESP, the number of reduce tasks has to be equal to number of reduce slots. It is unrealistic to configure either the same number of reduce tasks or the single wave of the reduce phase for all the jobs. It can be argued that in practice, the number of reduce tasks varies depending on the size of the input dataset, the type of a Hadoop application (e.g., CPU intensive, or disk I/O intensive) and user requirements. Furthermore, for the reduce phase, using multiple waves generates better performance than using a single wave especially when Hadoop processes a large dataset on a small amount of resources. While a single wave reduces the task setup overhead, multiple waves improve the utilization of the disk I/O.

Normally a Hadoop job execution is divided into a map phase and a reduce phase. The reduce phase involves data shuffling, data sorting and user-defined reduce functions. Data shuffling and sorting are performed simultaneously. Therefore, the reduce phase can be further divided into ashuffle (or sort) phase and a reduce phase performing userdefined functions. As a result, an overall Hadoop job execution work flow consists of a map phase, a shuffle phase and a reduce phase as shown in Fig. 1. Map tasks are executed in map slots at a map phase and reduce tasks run in reduce slots at a reduce phase. Every task runs in one slot at a time. A slot is allocated with a certain amount of resources in terms of CPU and RAM. A Hadoop job phase can be completed in a single wave or multiple waves. Tasks in a wave run in parallel on the assigned slots.

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering
(IJARMATE) Vol. 3, Special Issue 7, March 2017

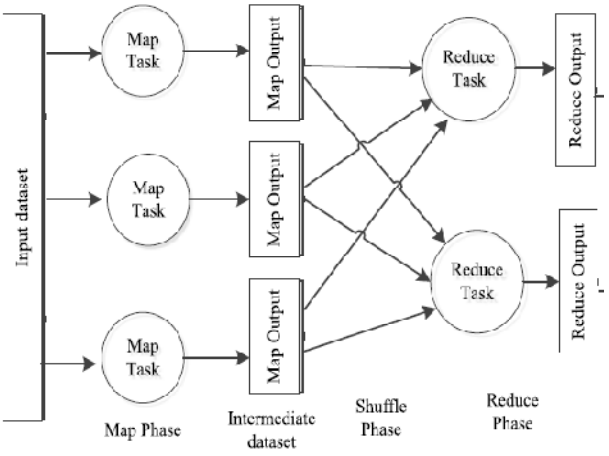


Fig. 1. Hadoop job execution flow

Modeling Map Phase

In this phase, a Hadoop job reads an input dataset from Hadoop distributed file system (HDFS), splits the input dataset into data chunks based on a specified size and then passes the data chunks to a user-define map function. The map function processes the data chunks and produces a map output. The map output is called intermediate data.

Modeling Reduce Phase

In this phase, a job reads the sorted intermediate data as input and passes to a user-defined reduce function. The reduce function processes the intermediate data and produces a final output. In general, the reduce output is written back into the HDFS.

Modeling Shuffle Phase

In this phase, a Hadoop job fetches the intermediate data, sorts it and copies it to one or more reducers. The shuffle tasks and sort tasks are performed simultaneously, therefore, generally consider them as a shuffle phase.

Hadoop performance modeling has become a necessity in estimating the right amount of resources for user jobs with deadline requirements. It should be pointed out that modeling Hadoop performance is challenging because Hadoop jobs normally involve multiple processing phases including three core phases (i.e. map phase, shuffle phase and reduce phase). Moreover, the first wave of the shuffle phase is normally processed in parallel with the map phase (i.e. overlapping stage) and the other waves of the shuffle phase are processed after the map phase is completed (i.e. non overlapping stage).

Objectives:

1. Offer the opportunities for Hadoop users to lease a certain amount of resources and pay for their use.

2. Provisioning mechanism to satisfy user jobs.

3. Accurately estimates job completion time and further provisions the required amount of resources for a job to be completed within a deadline.

4. Providing accuracy for performance of system.

The proposed system called Hadoop Performance Modeling for Job Optimization. In Proposed System it present improved HP model for Hadoop job execution estimation and resource provisioning. The improved HP work mathematically models all the three core phases of a Hadoop job. In contrast, the HP work does not mathematically model the non-overlapping shuffle phase in the first wave. The improved HP model employs Locally Weighted Linear Regression (LWLR) technique to estimate the execution time of a Hadoop job with the varied number of reduce tasks. In contrast, the HP model employs a simple linear regress technique for job

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering
(IJARMATE) Vol. 3, Special Issue 7, March 2017

execution estimation which restricts to a constant number of reduce tasks. Based on the job execution estimation, the improved HP model employs Langrage Multiplier technique to provision the amount of resources for Hadoop job to complete within a given deadline.

The major contributions of this system are as follows:

1. The improved HP work mathematically models all the three core phases of a Hadoop job. In contrast, the HP work does not mathematically model the non overlapping shuffle phase in the first wave.

2. The improved HP model employs Locally Weighted Linear Regression (LWLR) technique to estimate the execution time of a Hadoop job with a varied number of reduce tasks. In contrast, the HP model employs a simple linear regress technique for job execution estimation which restricts to a constant number of reduce tasks.

3. Based on job execution estimation, the improved HP model employs Lagrange Multiplier technique to provision the amount of resources for a Hadoop job to complete within a given deadline. The performance of the improved HP model is initially evaluated on an in-house Hadoop cluster and subsequently on Amazon EC2 Cloud.

The estimated values of both the shuffle phase and the reduce phase are used in the improved HP model to estimate the overall execution time of a Hadoop job when processing a new input dataset. Figure shows the overall architecture of the improved HP model, which summarizes the work of the improved HP model in job execution estimation. The boxes in gray represent the same work presented in the HP model. It is worth noting that the improved HP model works in an offline mode and estimates the execution time of a job based on the job profile.

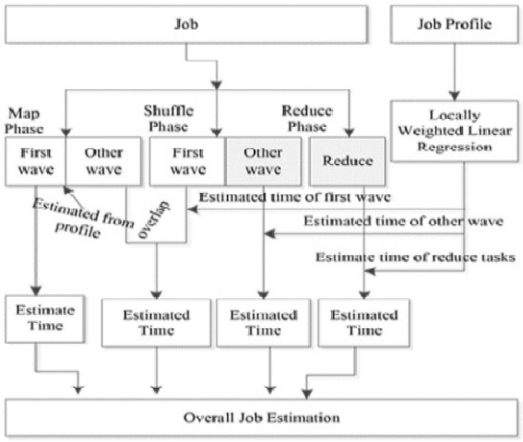


Fig. 2. System Architecture

PROPOSED ALGORITHM

Algorithm 1: Compute VM Load from data nodes

Input: ith Node input

Output: Idle or Normal Or Overloaded in percent

Compute Load (VM id) :

Weight Degree Inputs: The static parameter comprise the number of CPUs, the CPU dispensation speeds, the reminiscence size, etc. active parameters are the memory consumption ratio, the CPU exploitation ratio, the network bandwidth.

Procedure:

Step 1: Characterize a load limit set: F= {F1,F2. .Fm} with each Fire present the total number of the consideration.

Step 2: Calculate the load capacity as weight Load Degree(N)=(∑αi Fi) , Where, i=(1,...,m).

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering
(IJARMATE) Vol. 3, Special Issue 7, March 2017

Step 3: Ordinary cloud partition degree from the node consignment degree statics as: Load amount avg =∑(i=1..n)Load Degree(Ni)

Step 4: Three height node position are defined Load Degree(N)=0 for Inactive.

- 0 <Load Degree(N)<Load Degree(N) for overfull.

Load Degree(N)high ≤ Load Degree(N)for overloaded.

Algorithm 2: Equally Spread Current Execution Throttled Load balancing Algorithm

Input: File form user as Fi.

Output: Equally distributed chunks on data servers

Step 1: Read Fi from data owner with size

Step 2: count total number of data nodes Ni

Step 3: for each(score=read each vm node and call to computenode(k)) Read when k==null End for

Step 4: create data chunks base on server loads score.

Step 5: save all data on data nodes.

3.1 MODULE DESCRIPTION

There are four modules designed to implement the project.The four modules are:

1. Formation of clusters using Hadoop distributed file System.

2. Establishing Mapreduce function for the datasets.

3. Estimating job Execution Time using LWLR

4. Resource Provisioning using LMT.

4. Performance Analysis.

FORMATION OF CLUSTERS USING HADOOP DISTRIBUTED FILE SYSTEM

Setting up the Hadoop cluster in Hadoop distributed file system with the use of data node, name node. Secondary name node, job tracker, task tracker to perform Hadoop operations using mapreduce algorithm. Here individual nodes perform its own operations . Data node stores the data in Hadoop file system. Name node keeps the directory of all files in the file system. Multinode clusters are formed to perform the tasks for large datasets. A typical file in HDFS is gigabytes to terabytes in size. It should support tens of millions of files in a single instance. The nodes of clusters are initially formed in this module.

ESTABLISHING MAPREDUCE FUNCTION FOR THE DATASETS

The workload is classified and given to the mapreduce framework for MAP and REDUCE process. It splits the dataset into individual lines by using mapper instances and schedule the jobs inaccordance with hadoop cluster components.Workload undergoes splitting, shuffling, sorting and reducing operations. Mapreduce allows for distributed processing of the map and reduction operations. Provided that each mapping operation is independent of the others, all maps can be performed in parallel,it is limited by the number of independent data sources .

The Mapreduce algorithm is a two-step method for processing a large amount of data. It process the parallel problems across huge datasets. These frameworks supports up to petabytes of data.

Map step: The dataset is given as a client input into hadoop distributed file system.Under the map process, the given dataset is splitted into individual lines or words using mapper instances and an intermediate output is obtained by output.

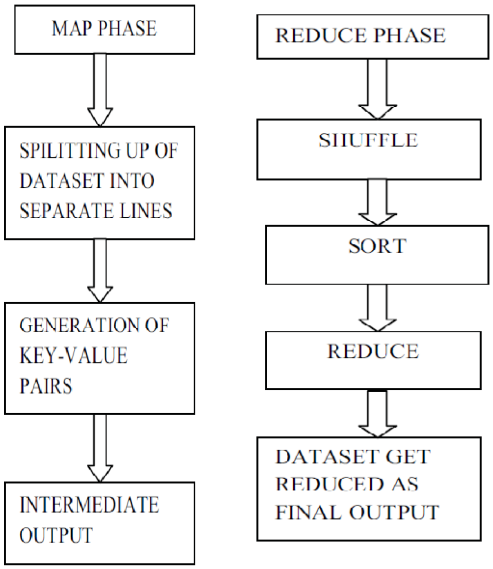
ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering
(IJARMATE) Vol. 3, Special Issue 7, March 2017

Reduce step: Shuffle,sort and reduce are the three process to be done in reduce phase. The intermediate output obtained in map phase undergoes shuffle, and it gets sorted order. At the final phase reduce ,the dataset get reduced. Each step starts only after the previous step is completed.

ESTIMATING JOB EXECUTION TIME USING LWLR

LWLR maintains two separate queues for current running jobs and waiting jobs. By default, incoming jobs enter the waiting queue. The scheduling optimizer determines which job is to be moved to the running queue based on the solution of the LMT. For example, low priority jobs or jobs with distant deadlines may not be immediately allocated resources by the LMT algorithm, i.e., $uj(t) = 0$. These jobs wait until they receive resource allocation from the scheduling optimizer. Since the resource allocation is determined once for every control interval, it is possible that short jobs whose deadline is earlier than the next control interval may miss their deadlines due to the late allocation of resources. To this end, we provide a fast path for short jobs in the job queue management, that is, it immediately moves a short job to the running queue.



Fuzzy performance model takes allocated resources and job size as inputs and generates the estimated job completion time as outputs. The model is updated periodically based on the measured job progress at each control interval.

RESOURCE PROVISIONING USING LMT

Resource predictor takes the history information on resource availability and predicts the amount of available resources for the next few intervals. Scheduling optimizer adjusts the number of slots allocated to each running job based on an online receding horizon control algorithm. However, a job's completion time yj can only be measured when the job finishes. It is often too late for the Hadoop scheduler to intervene if a job already misses its deadline. To this end, break down a job's execution into small intervals and apply calibrated deadlines for each interval. Consider a job's deadline is 100 minutes away and the execution is divided into 10 intervals. If the job can finish one tenth of the total work in each interval, it can meet the overall

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering
(IJARMATE) Vol. 3, Special Issue 7, March 2017

deadline. The Hadoop scheduler can adjust the resource allocation if a job's execution is considered slow based on its progress on individual intervals. Such a breakdown of job execution also allows the scheduler to look forward into future intervals and apply optimization considering future resource availability.

PERFORMANCE ANALYSIS THROUGH MAKE SPAN AND RESPONSE TIME FACTORS

To identify the performance analysis of the disks located in Hadoop distributed file system by identifying its computation time and its response factors. The analysis dependent on parameters such as: dataset size, number of nodes, number of reducers and loading overhead. The results indicate strong dependence on the amount of reducers and IO performance of the cluster, which proves the common opinion that Mapreduce is IO bound. These results can help to compare performance behavior of different languages and serve as a basis for understanding the influence of configuration parameters on the final performance.

4. CONCLUSION

The improved HP model mathematically modeled three core phases i.e. map phase, shuffle phase and reduce phase included overlapping and nonoverlapping stages of a Hadoop job. The proposed model employed LWLR to estimates execution duration of a job that takes into account a varied number of reduce tasks The LWLR model was validated through 10-fold cross-validation technique and its goodness of fit was assessed using R-Squared. In future for resources provisioning, the model applied Lagrange Multiplier technique to provision right amount of resources for a job to be completed within a given deadline. The improved HP model was more economical in resource provisioning than the HP model.

FUTURE ENHANCEMENT

Running a MapReduce Hadoop job on a public cloud such as Amazon EC2 necessitates a performance model to estimate the job execution time and further to provision a certain amount of resources for the job to complete within a given deadline. It has presented an improved HP model to achieve this goal taking into account multiple waves of the shuffle phase of a Hadoop job. The improved HP model was initially evaluated on an in-house Hadoop cluster and subsequently evaluated on the EC2 Cloud. The experimental results showed that the improved HP model outperforms both Starfish and the HP model in job execution estimation. Similar to the HP model, the improved HP model provisions resources for Hadoop jobs with deadline requirements. However, the improved HP model is more economical in resource provisioning than the HP model.Both models over-provision resources for user jobs with large deadlines in the cases where VMs are configured with a large number of both map slots and reduce slots.

One future work would be to consider dynamic overhead of the VMs involved in running the user jobs to minimize resource over-provisioning. Currently the improved HP model only considers individual Hadoop jobs without logical dependencies.

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering
(IJARMATE) Vol. 3, Special Issue 7, March 2017

Another future work will be to model multiple Hadoop jobs with execution conditions.

REFERENCES

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

[2] R. L€ammel, "Google's MapReduce programming model—Revisited," Sci. Comput. Programm., vol. 70, no. 1, pp. 1–30, 2008.

[3] B. He, W. Fang, Q. Luo, N. K. Govindaraju, and T. Wang, "Mars: A MapReduce framework on graphics processors," in Proc. 17th Int. Conf. Parallel Archit. Compilation Techn., 2008, p. 260.

[4] K. Taura, T. Endo, K. Kaneda, and A. Yonezawa, "Phoenix: A parallel programming model for accommodating dynamically joining/ leaving resources," ACM SIGPLAN Notices, vol. 38, no. 10, pp. 216–229, 2003.

[5] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: Distributed data-parallel programs from sequential building blocks," ACM SIGOPS Oper. Syst. Rev., vol. 41, no. 3, pp. 59–72, Mar. 2007.

[6] Apache Hadoop [Online]. Available: http://hadoop.apache.org:/ (Last accessed: 1 March 2015).

[7] D. Jiang, B. C. Ooi, L. Shi, and S. Wu, "The performance of Map- Reduce: An in-depth study," Proc. VLDB Endowment, vol. 3, nos. 1/2, pp. 472–483, Sep. 2010.

[8] U. Kang, C. E. Tsourakakis, and C. Faloutsos, "PEGASUS: Mining Peta-scale Graphs," Knowl. Inf. Syst., vol. 27, no. 2, pp. 303–325, May 2011.

[9] B. Panda, J. S. Herbach, S. Basu, and R. J. Bayardo, "PLANET: Massively parallel learning of tree ensembles with MapReduce," Proc. VLDB Endowment, vol. 2, no. 2, pp. 1426–1437, Aug. 2009.

[10] A. Pavlo, E. Paulson, and A. Rasin, "A comparison of approaches to large-scale data analysis," in Proc. ACM SIGMOD Int. Conf. Manag. Data, 2009, pp. 165–178.

[11] X. Lin, Z. Meng, C. Xu, and M. Wang, "A practical performance model for Hadoop MapReduce," in Proc. IEEE Int. Conf. Cluster Comput. Workshops, 2012, pp. 231–239.

[12] X. Cui, X. Lin, C. Hu, R. Zhang, and C. Wang, "Modeling the performance of MapReduce under resource contentions and task failures," in Proc. IEEE 5th Int. Conf. Cloud Comput. Technol. Sci., 2013, vol. 1, pp. 158–163.

[13] V. Jalaparti, H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron. Bazaar: Enabling predictable performance in datacenters. Microsoft Res., Cambridge, U.K., Tech. Rep. MSR-TR- 2012–38 [Online].