Robust MapReduce Scheduling in shared Environment

[1]Devika Rani B J [2] Sandhya G [3]Sindhu E [4] Mr. Sreenivasa B R
[1,2,3] UG Student, Dept. of CSE,RRCE.
[4]Assistant Professor, Dept. of CSE,RRCE.

***Abstract*— In this paper we present a MapReduce task scheduler for shared environments in which MapReduce is executed along with other resource-consuming workloads. All workloads may potentially share the same data store, some of them consuming data for systematic analysis purposes while others acting as data generators. This category of scenario is becoming increasingly important in data centers where improved resource utilization can be achieved through workload unification ,and is specially challenging due to the interaction between workloads of different nature that compete for limited resources. The proposed scheduler aims to improve resource utilization while observing overall time goals. Not like other MapReduce schedulers this approach takes the resource demands for non-MapReduce workloads, and assumes that the efficient amount of resources made available to the MapReduce applications is liable over time. In our proposal improves the management of MapReduce jobs in the presence of variable resource availability, rise in the accuracy of the estimations made by the scheduler, thus improving overall time goals without an influence on the fairness of the scheduler.**

Keywords*— *MapReduce, Scheduling, Distributed, Analytics, Transactional, Adaptive, Availability, Shared Environment

I. INTRODUCTION

In recent years, the industry and research community have witnessed an extraordinary gr0owth in research and development of data-related technologies. In addition to distributed, large-scale data processing workloads such as MapReduce, other distributed systems have been introduced to deal with the management of huge amounts of data providing at the same time support for both data-analytics and transactional workloads.

Instead of running these services in completely dedicated environments, which may lead to underutilized resources, it is becoming more common to multiplex different and complementary workloads in the same machines. This is turning clusters and data centers into shared environments in which each one of the machines may be running different applications simultaneously at any point in time: from database servers to MapReduce jobs to other kinds of applications. This constant change is challenging since it introduces higher variability and thus makes performance of these systems less predictable.

In particular, proposed system considered an environment in which data analytics jobs, such as MapReduce applications, are collocated with transactional workloads. In this scenario, deep coordination between management components is critical, and single applications cannot be considered in isolation but in the full context of mixed workloads in which they are deployed. Integrated management of resources in presence of MapReduce and transactional applications is challenging since the demand for transactional workloads is known to be bursty and varying over time, while MapReduce schedulers usually expect that available resources are unaltered over time. Transactional workloads are usually of higher priority than analytics jobs because they are directly linked to the QoS perceived by the users. As such, in our approach transactional workloads are considered as critical and we assume that only resources not needed for transactional applications can be committed to MapReduce jobs.

II. MOTIVATION

Consider a system running two major distributed frame- works: a MapReduce deployment used to run background jobs, and a distributed data-store that handles transactional operations and serves data to a front-end. Both workloads share the same machines, but since the usage of the frontend changes significantly over time depending on the activity of external entities, so does the availability of resources left for the MapReduce jobs. Notice that the demand of resources over time for the front-end activities is supposed to be well characterized, and therefore it can be predicted in the form of a given function f(t) known in advance.

In the proposed system, the MapReduce workload consists of 3 identical jobs: J1, J2, and J3. All jobs are submitted at time 0, but have different deadlines: D1 (6.5h), D2 (15h), and D3 (23.1h). Collocated with the MapReduce jobs, we have a front-end driven transactional workload that consumes available resources over time. The amount of resources committed to the critical transactional workload is defined by the function f(t).

Fig 1 shows the expected outcome of an execution using a MapReduce scheduler that is not aware of the dynamic availability of resources and thus assumes resources remain stable over time. Fig 2 shows the behavior of a scheduler aware of changes in availability and capable of leveraging the characteristics of other workloads to scheduler MapReduce jobs. In both Figures, the solid thick line represents f(t). Resources allocated to the transactional workload are shown as the white region on top of f(t), while resources allocated to the MapReduce workload are shown below f(t), being each job represented by a different pattern. X-axis shows time, while Y-axis represents compute nodes allocated to the workloads.

Fig 1 represents the expected behavior of a scheduler that is not aware of the presence of other workloads. As it is not

able to predict a future reduction in available resources, it is not able to cope with dynamic availability and misses the deadline of the first two jobs because it unnecessarily assigns tasks from all jobs (e.g. from time 0 to 5, and from time 7 to 11 approximately). On the other hand, Fig 2 shows the behavior of the scheduler proposed in this paper, the Reverse-Adaptive Scheduler, which distributes nodes across jobs considering future availability of resources. From time 0 to D1, most of the task trackers are assigned tasks from J1, and the remaining to J2 since it also needs those resources to reach its goal on time. From time D1 to D2, most of the resources go to J2 in order to meet a tight goal. However, as soon as J2 is estimated to reach its deadline, a few tasks from J3 are assigned as well starting around time 4. Finally, from time D2 until the end only tasks from J3 remain to be executed.

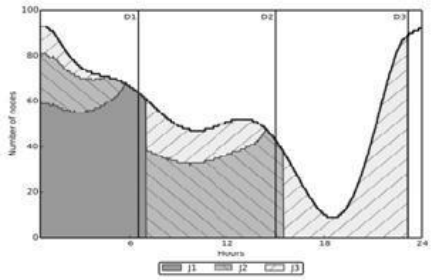

Fig1: Distribution of assigned resources overtime running the sample workload using the adaptive scheduler [6] without dynamic resource availability

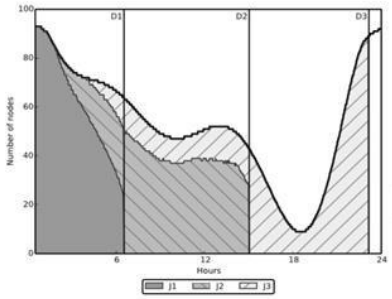

Fig 2: Distribution of assigned resources over time running the sample workload using the Reverse-Adaptive Scheduler

III. RELATED WORK

Much work has been done in the space of scheduling for MapReduce. Since the number of resources and slots in a Hadoop cluster is fixed throughout the lifetime of the cluster, most of the proposed solutions can be reduced to a variant of the task-assignment or slot-assignment problem. The Capacity Scheduler [12] is a pluggable scheduler developed by Yahoo! which partitions resources into pools and provides priorities for each pool. Hadoop's Fair Scheduler [13] allocates equal shares to each tenant in the cluster. All these schedulers are built on top of the same resource model and do neither support high-level goals nor dynamic availability in shared environments.

The performance of MapReduce jobs has attracted much interest in the Hadoop community. Recently, there has been increasing interest in user-centric data analytics. The Adaptive Scheduler [6] enables soft-deadline support for MapReduce jobs. It differs from this paper's proposal in that it does not take into consideration neither the resources of the system nor other workloads. Similarly, Flex [14] is a scheduler proposed as an add-on to the Fair Scheduler to provide Service-Level-Agreement (SLA) guarantees. More recently, Aria [8] introduces a novel resource management framework that consists of a job profiler, a model for MapReduce jobs and a SLO- scheduler based on the Earliest Deadline First scheduling strategy. Flex and Aria are both slot-based and therefore suffer from the same aforementioned limitations.

New platforms have been proposed to mix MapReduce frameworks like Hadoop with other kinds of workloads. Mesos [15] intends to improve cluster utilization on shared environments, but is focused on batch-like and HPC instead of transactional workloads. Finally, the Hadoop community has also recognized the importance of developing a resource-aware scheduling for MapReduce, and proposed a framework [16] that introduces a resource model consisting of a 'resource container' which is fungible across jobs. We think that our resource management techniques can be leveraged within this framework to enable better resource management

IV. DESIGN

System architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

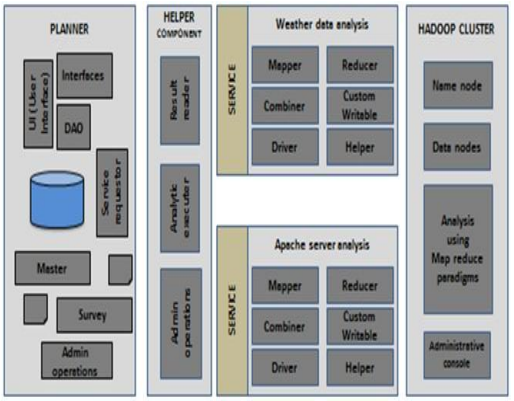

Fig 3: System Architecture for MapReduce

System architecture consists of planner interface, helper component and the hadoop cluster.

Planner interface interacts with user to register and schedule their jobs. Planner consists of the following:

i. Master planner
ii. Survey planner
iii. DAO-Data Access Object,
iv. Service requestor.

DAO has all the user registered jobs that has to be executed.

Master planner communicates with DAO to know whether any job is to executed, it runs for every 6 seconds.

Survey planner checks the DAO for every 3 seconds to execute the missed or failed jobs. Service requestor will speak

to helper component to execute the job, whenever the job has to be executed

Hadoop cluster has –

i. Name nodes
ii. Data nodes

A name node stores the meta-data and it runs on high quality hardware. Name node is the master component which is point of contact to external clients. It will be controlling the data nodes. Data nodes are the slave components which work as per the name node instructions; this is the place where actual data is stored. It is responsible for serving read and writes requests for the clients each data node holds a part of the overall data and processes the part of the data which it holds.

DATA FLOW

Data flow diagram is a graphical representation of the flow of data through information system, modeling its process aspects.

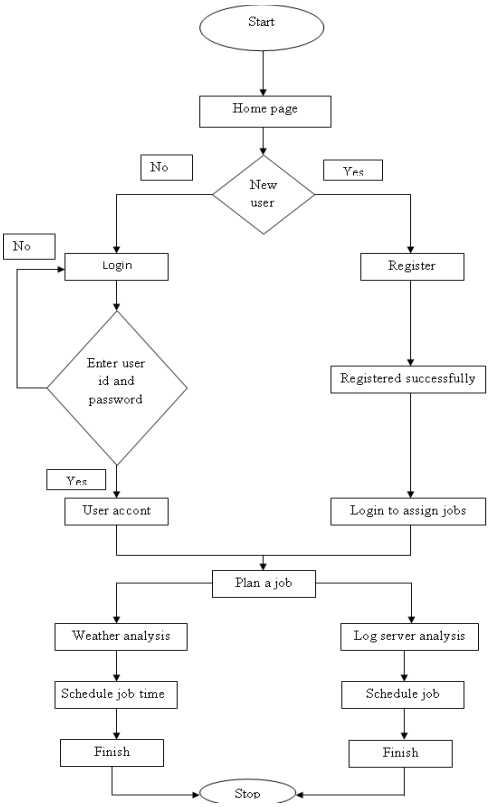

Fig 4: Data flow diagram of user credentials

The flow is described as follows:

When the program is executed the user is directed to the home page which has multiple options for user. If the user is a new user, user has to register himself by giving his name and password etc., if the user is already registered he can login using is login credentials. After logging in he can schedule or plan the jobs as per his needs he can run the job weekly, monthly, daily. After scheduling his job is scheduled to run and process the job. The results can be viewed in the browser or he can redirect it to his mail. According to his schedule he will get the notifications.

V.IMPLEMENTATION

We are implementing two modules namely MapReduce paradigm and analysis. In analysis two types of analysis is done namely apache log server analysis and weather data analysis.

MAPREDUCE PARADIGM: MapReduce is a programming model/tool in Hadoop used to process the data stored in HDFS.

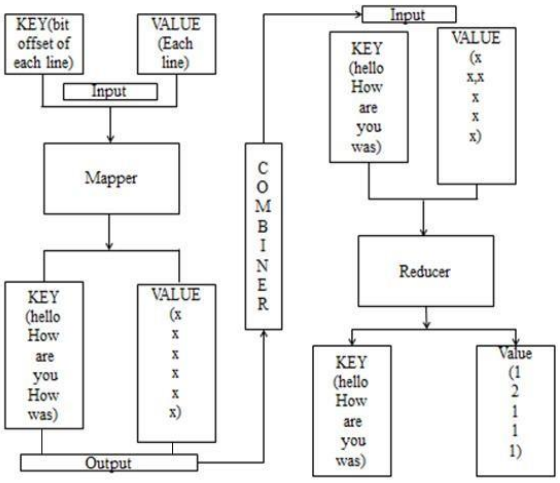

Fig 5: MapReduce program for word count example

Reporter and combiner is provided by default in Hadoop. Input is given by the reporter to the mapper component Mapper component has map(). The input to the mapper component is a (key, value) pair, where key is the bit offset of the each line and value is the complete sentence or line. Map() executes separately for each line. Output of the mapper is also a (key, value) pair where key is words of the line spitted based on the space and value is just a count or the variable representing that word. This output is passed to the combiner where combiner will eliminate the duplicate keys present in the output of the mapper and gives the output of the mapper as input to the reducer in the form of (key, value) pair. The reducer retains the key and the value gives the added value of the key like how many times the same word does had occurred in the entire document. Using this concept we can do the analysis of log details of servers like Face book, college websites, and twitter also the temperature details like maximum and minimum temperature of the given input.

APACHE LOG ANALYSIS: The server log file (like FB or twitter or college website etc.) is given as input and to know how many people where active on particular day at specific time .Using the result one can make the availability of the website more or less based on the analysis.
For e.g., if there is a more access to ABC website at 2pm every day, then they can make the availability of that website more at that particular time on every day.

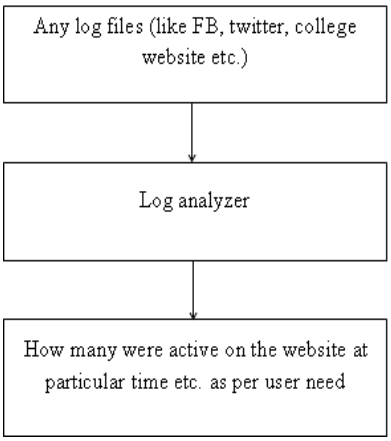

Fig 6: Log analyzer

The result of this analysis can be used to give advertisements to its particular users of their interests and also the companies can double their servers to make the availability of the website more.

WEATHER DATA ANALYSIS: The user specifies the input of various cities with temperature date and time, where the output of the analysis will be the city with maximum and minimum temperature. The user gets the weather input from the weather sensors.

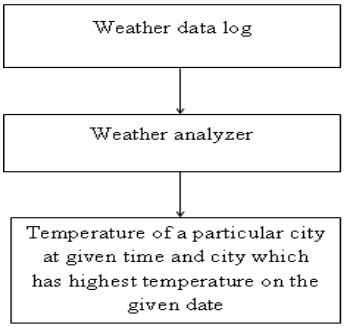

Fig 7: Weather data analyzer

The user can view the results in his account or he can redirect it to his email where he can view whenever he wants.

RESULTS

This concept can be used in transactional process to know the user details. For example, a manager can know the transaction details of the particular bank like how many transactions are done in a single day at a particular hour etc.

ACKNOWLEDGMENT

We express our heartfelt thanks to **Dr.UshaSakthivel**, Professor & Head, Department of Computer Science and Engineering, Rajarajeswari college of Engineering, for her inspiration. we would also like to thank our project guide **Mr.Sreenivasa B R**, Assistant Professor, Department of Computer Science and Engineering,RajaRajeswari College of Engineering, Bangalore, under whose able guidance this project work has been carried out and completed successfully.we also thank our beloved principal and management for their co-operation and support, for providing a very good infrastructure and all the kindness forwarded to us in carrying out this project in college.

CONCLUSION

In this paper we have presented the Reverse-Adaptive Scheduler, which introduces a novel resource management and job scheduling scheme for MapReduce when executed in shared environments along with other kinds of workloads. Our scheduler is capable of improving resource utilization and job performance. The model we introduce allows for the formulation of a placement problem which is solved by means of a utility-driven algorithm. This algorithm in turn provides our scheduler with the adaptability needed to respond to changing conditions in resource demand and availability of resources.

In this project, we have explored the solution to big data problem using hadoop data cluster, HDFS and MapReduce programming framework using big data prototype application scenarios. The results obtained from various experiments indicate favorable results of above approach to address big data problem. Future work will focus on performance evaluation and modeling of hadoop data-intensive applications on cloud platforms like Amazon Elastic Cloud(EC2).

REFERENCES

[1] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," in OSDI'04: Proceedings of the 6th Symposium on Operating Systems Design and Implementation. San Francisco, CA: USENIX Association, December 2004, pp. 137–150.

[2] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber, "Bigtable: A distributed storage system for structured data," ACM Trans. Comput. Syst., vol. 26, no. 2, pp. 4:1–4:26, Jun. 2008.

[3] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and
W. Vogels, "Dynamo: amazon's highly available key-value store," in Proceedings of twenty- first ACM SIGOPS symposium on Operating systems principles, ser. SOSP '07. NY, USA: ACM, 2007, pp. 205–220.

[4] L. A. Barroso, J. Clidaras, and U. H¨olzle, "The datacenter as a computer: An introduction to the design of warehouse- scale machines, second edition," Synthesis Lectures on Computer Architecture, vol. 8, no. 3, pp. 1–154, 2013.

[5] G. Pacifici, W. Segmuller, M. Spreitzer, and A. Tantawi, "Dynamic estimation of cpu demand of web traffic," in Proceedings of the 1st international conference on Performance evaluation methodolgies and tools, ser. valuetools '06. New York, NY, USA: ACM, 2006.

[6] J. Polo, D. Carrera, Y. Becerra, M. Steinder, and I. Whalley, "Performance-driven task co-scheduling for MapReduce environments," in Network Operations and Management Symposium, NOMS, Osaka, Japan, 2010, pp. 373–380.
[7] J. Polo, C. Castillo, D. Carrera, Y. Becerra, I. Whalley, M. Steinder, J. Torres, and E. Ayguad´e, "Resource-aware adaptive scheduling for mapreduce clusters," in Middleware 2011, ser. Lecture Notes in Computer Science, vol. 7049. Springer Berlin Heidelberg, 2011, pp. 187–207.
[8] A. Verma, L. Cherkasova, and R. H. Campbell, "ARIA: Automatic Re- source Inference and Allocation for MapReduce Environments," in 8th IEEE International Conference on Autonomic Computing, Karlsruhe, Germany., June 2011.
[9] D. Gmach, J. Rolia, L. Cherkasova, and A. Kemper, "Workload anal- ysis and demand prediction of enterprise data center applications," in Workload Characterization, 2007. IISWC 2007. IEEE 10th International Symposium on, Sept 2007, pp. 171–180.
[10] C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, "A scalable application placement controller for enterprise data centers," in Procs.of the 16th intl. conference on World Wide Web, ser. WWW '07. NY, USA: ACM, 2007, pp. 331–340.
[11] Y. Chen, A. S. Ganapathi, R. Griffith, and R. H. Katz, "A methodology for understanding mapreduce performance under diverse workloads," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2010-135, Nov 2010.
[12]Yahoo! Inc. Capacity Scheduler. http://developer.yahoo.com/blogs/hadoop/posts/2011/02/capac ity-scheduler/.
[13] M. Zaharia, A. Konwinski, A. D. Joseph, R. Katz, and I. Stoica, "Improving mapreduce performance in heterogeneous environments," in OSDI'08. Berkeley, USA: USENIX Association, 2008, pp. 29–42.
[14] J. Wolf, D. Rajan, K. Hildrum, R. Khandekar, V. Kumar, S. Parekh, K.- L. Wu, and A. Balmin, "Flex: A slot allocation scheduling optimizer for mapreduce workloads," in Middleware 2010, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2010, vol. 6452, pp. 1–20.
[15] B. Hindman, A. Konwinski, M. Zaharia, A. Ghodsi, A. D. Joseph, R. Katz, S. Shenker, and I. Stoica, "Mesos: A platform for fine-grained resource sharing in the data center," in 8th USENIX conference on Networked systems design and implementation. USENIX Association, 2011, pp. 22–22.
[16] Arun Murthy. Next Generation Hadoop.[Online]. Available:http://developer.yahoo.com/blogs/hadoop/posts/201 1/03/mapreduce-