A Reactive Scheduling Strategy Applied On MapReduce OLAM Operators System

Shengbing Ren
School of Software, Central South University, Changsha, P.R China
rsb@csu.edu.cn

Dieudonne Muheto
School of Information Science and Engineering, Central South University, Changsha, P.R China
muheto184@gmail.com

***Abstract*—The combination of Data warehousing and data analysis techniques such as OLAP (Online Analytic Processing) and data mining through the Hadoop framework is an innovative way to treat large volumes of data. However, this way poses serious scheduling and combining tasks issues that bring more challenges.**
In this paper, we propose strategies to answer these questions, namely parallel OLAM (Online Analytic Mining) MapReduce Operators and a Reactive Scheduling Policy. OLAM MapReduce Operators divide jobs into two parts, the first includes all the operators that are used to create an OLAM CUBE and the second includes those who exploit the cube by data mining algorithms. The proposed policy coordinates the workflow generated by these operators, relying on model-based events. Our simulation experience shows that our strategy has a cumulative force that it reduces the execution time of the entire cluster at each request.

***Index Terms*—OLAP, data mining, parallel OLAM, MapReduce Operators, Reactive Scheduling Policy**

I. INTRODUCTION

Data warehouses are an effective solution for managing large data volumetric. Online Analysis completes data warehouses by providing tools for visualization, structuring and exploration of data cubes in order to discover relevant information. On the other hand, data mining uses techniques such as description, classification and explanation to induce knowledge models.

The idea of combining on-line analysis and data mining [1, 2] is an effective solution to strengthen the process of decision support. Indeed, there are two areas that can be completed as part of a unified analysis process.

With these systems, there is integration problem of OLAP (Online Analytic Processing) tasks and data mining algorithms. In addition, the time cost for the treatment of these operations is substantial, and that is why we propose to create parallel OLAM (Online Analytic Mining) operators using MapReduce Paradigm. These operators are implemented using the Hadoop framework.

In the remainder of this paper, we describe in section II background for a proper understanding of the contribution of this work. We define in Section III the parallel OLAM operators that are coupling OLAP and DM as well as the timely processing of requests decision. Section IV deals with the efficient scheduling of these operators under certain constraints such as the multiple use of the system by many users. And finally we have a representative section IV summarizes the work done in the field, and Section V gives a conclusion and a line of future work.

Manuscript received January 1, 2012; revised October 25, 2012; accepted November 1, 2012.

II. BACKGROUND

Here, we describe the concepts relative to OLAM, and Hadoop the framework which we use to implement an OLAM system as a distributed system.

A. OLAM Technologies

OLAM (also called OLAP Mining) is the coupling of OLAP with data mining in multidimensional databases (data warehouses) [1, 2]. We can define a data warehouse as a dedicated database to store the set of used data in the context of decision making and decision analysis. The Data warehouse is exclusively reserved for this purpose. According to Bill Immon [3], the father of the concept, the data warehouse has four characteristics:

- Subject Oriented: At the heart of the data warehouse, data is organized by theme. Data specific to a theme, such sales will be repatriated from different OLTP (Online transaction Processing) production databases and combined.
- Integrated: Data from heterogeneous sources using different type of format must be integrated before offered for use.
- Non volatile: The data do not disappear and do not change over the treatment over time(Read-Only).

doi:10.4304/jsw.7.11.2649-2656

- Historicized: Non-volatile data are time-stamped. We can thus visualize the evolution in the time of a given value. The detail of the archive is of course on the nature of the data. All data does not deserve to be archived.

It is supplied with data from production databases via ETL (Extract Transform Load) tools. It is at this level that data warehouse involves data cleaning, data integration and data transformation. Data warehouses generalize and consolidate data with multidimensional model [4,5].

Data Warehouse works with OLAP tools for interactive analysis of multidimensional data of varied granularities, which facilitates effective data mining. Many other data mining functions, such as association, classification, prediction, and clustering, can be integrated with OLAP operations to enhance interactive mining of knowledge at multiple levels of abstraction [6, 7, 8].Therefore, data warehousing and OLAP form an essential step in the knowledge discovery[9,10] process. As architecture, we have three-tier architecture:

- A data warehouse database server as the bottom tier.
- The middle tier is a server that contains an OLAP engine and Data Mining engine.
- The top tier is a front-end client layer, which contains query and reporting tools.

Data Cube Computation is an essential task in data warehouse implementation. To understand this concept we go in giving a series of definitions.

- A cube is an abstract representation of information organized in set of dimensions. A dimension is a line of analysis that is a basis of which the data will be analyzed, e.g. time.
- A measure (or variable) is the data element that is analyzed. A measure can have from one to several thousands of values.
- A fact represents a value of a measurement according to one member of each dimension.

Dimensional models such as Star schema and Snowflake schema are the foundation of OLAP Cube Construction. These schemas are represented as a single facts table surrounded by dimensions. For the star model, any multi-level dimension is flattened in one dimension and for the snowflake schema, for any multi-level dimension at least one dimension level is managed in a separate structure from the other levels.

B. Hadoop Framework

Hadoop is an open source project coordinated by the Apache Software Foundation, whose purpose is to implement a distributed execution environment [11, 12]. Hadoop is not a monolithic project but a set of sub-projects. Here we describe briefly the two main sub-projects.

- HDFS (Hadoop Distributed File System): Hadoop is designed to handle massive amounts of data. Therefore the usefulness of HDFS makes sense, especially when the set of data to be processed exceeds the storage capacity of a single machine. In HDFS, the files are partitioned in blocks.

An HDFS cluster consists of two types of major components [13]: a Name node and several Data nodes. The Name node is the cornerstone of the file system. He manages the namespace and the file system tree, metadata, files and directories. Without Name node, all files can be considered lost because there would be no way to restore files from the blocks. There is only one instance of Name node by HDFS Cluster. The Data nodes store and restore the data blocks. They also periodically communicate to Name node the list of blocks they host.

- MapReduce is a programming paradigm introduced by Google [14] to handle large volumes of data. The model is relatively simple. It consists of cutting treatment in two phases: the first phase (Map) is a step of ingestion and processing of data and the second phase (Reduce) is a step of melting the recordings by key to form the final result. The input data to be processed are cut into small units, each treated in parallel by a Map function. The treatments results are sorted by key unit to form data units passed to a function Reduce. This is why the MapReduce programs are inherently designed to be executed in parallel and it is possible to distribute processing.

A MapReduce Job is a processing unit implementing a set of data in entry, a MapReduce program and configuration items. Running a MapReduce job requires a sharing of responsibility between development elements (dataset, MapReduce program, Hadoop configuration files) and infrastructure components (Hadoop cluster). A MapReduce cluster consists of components used to control the process of running a job: a Job Tracker and several Task Trackers. The job tracker coordinates the execution of jobs across the cluster. It communicates with task trackers by assigning tasks execution (map or reduce).

Finally, a Hadoop cluster is typically organized in master/slave and cohabited agents (or services) [13] as follows: elements running on the master node(Job Tracker, Name Node) and elements executed on the slave nodes (Data Nodes and Task Trackers). The communication between nodes is performed by Remote Procedure Call (RPC).

III. PARALLEL OLAM OPERATORS SYSTEM

The problem here is the effective interaction between OLAP and Data Mining. Combining these two techniques leads us to think intuitively Cubing then Mining or Mining then Cubing. There are countless ways to do this work as shown in [15, 16, 17, 18].

The design of a system depends, in general, aims to achieve. OLAP provides a means for users to navigate in multidimensional data in order to discover relevant information. However, this exploratory approach requires sufficient expertise of the user allowing him to extract the knowledge the most interesting in terms of its scope of analysis.

In this process, the user is manually discovers by using the OLAP tools, the potential knowledge contained in data cubes. Indeed, OLAP technology offers

opportunities to see the facts, but does not describe the significance or possible relationships between these facts.

It is in this sense that we propose a model for our OLAM system in Fig. 1, which will allow us:

- To classify or group the facts in order of proximity, not just along lines of analysis.
- To explain the implications or associations between facts, not just to explore the facts.

From the Fig. 1 we can see that there are operators, and are highlighted by arrows. All these operators are parallelized and that is the subject of the next section.

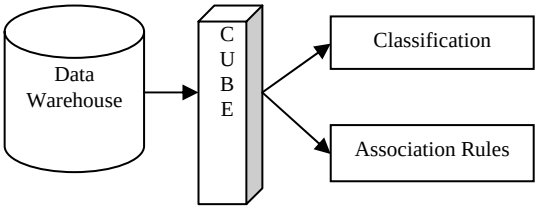

Figure1. Classification and Association rules from cube

A. Parallelization Strategy

The processing performed by these operators are parallisable through MapReduce parallel programming model. Note that MapReduce is a technique that breaks down the processing of an operation (Hadoop Job) in several stages, including two elementary to optimize parallel processing of data. The various operations like Mapping, Shuffling and Reducing manipulate the lists of outputs and inputs in the form <Key, value>.

The problem of data representation is an important issue in the coupling between the online analysis and data mining. Indeed, on one side, the mining algorithms can only operate on that data presented in the form of a classical attribute-value table. On the other side, in the context of a data warehouse, data is organized in a multidimensional structure suitable for online analysis.

With all these issues, we are trying to build a strategy adapted to MapReduce procedures and objectives of our model cited above. Thus classify all operators into two groups: the first that will allow us to create an OLAM Cube [19] and the latter that will be applied to this structure. An OLAM Cube is a cube that can be used as input data for data mining algorithms.

A data cube is an abstract representation of multidimensional data. It corresponds to the fact table in the schema of a warehouse. It is also called hypercube, which is as its name suggests, a multidimensional array of values representing the distribution of events according to two variables (dimensions) or more. Of the beginning of its creation, the data cube has been proposed as the result of a query combining the GROUP-BY in all combinations of dimensions [20]. The result of a single GROUP-BY is a cuboid.

Thus our new model is as Fig. 2. With all these difficulties we apply known techniques such as binarization and the discretization is that will allow us to find tables compatible with the data mining algorithms. The general idea of these techniques is to transform input data into discrete attributes, but first let's make sure that the data cubes are in tabular form.

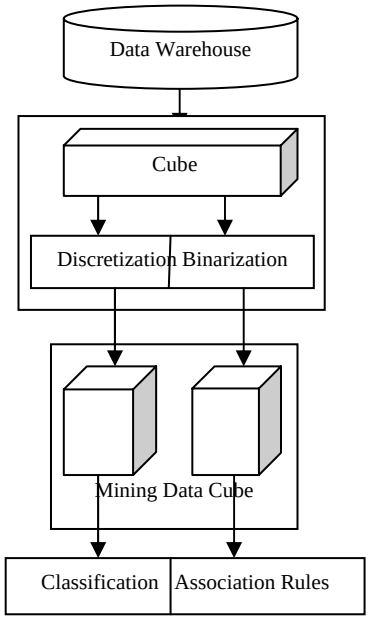

Figure2. OLAM System Model

B. Pipelines Applications

OLAM task as seen above is a collection of multiple other operations. With an OLAM engine over Hadoop, it is up to transform these operators in MapReduce operations. All these operations are not done in a single process but several MapReduce procedures. There is a constant almost obvious that all these operations are done in a chain of MapReduce jobs. These flows are in fact mapreduces data flows. The entry point of a MapReduce is a mass of data composed of several files. These files will be cut into splits. At the initialization, the number of maps depends directly on the number of splits calculated beforehand. The path of the data flow is based on the part Map and part Reducer. A map task traverses the split which is associated launch function: map (key, value) for each of the different key split. After this phase, each of the key result is sent to an OutputCollector. A combiner function can be called (for each map before writing to disk) and will combine for each key all the associated values. This call can reduce the output of a map and there will be less data to write on the disc. The maps output files are on the local disk of the node (tasktraker). The reducers need these files as input for processing. The tasks of Reduce are going to get the data output tasks map. To do this, Reduce tasks need to know the progress of map tasks to begin treatment. Upon receipt of a request, the JobTracker instantiate an object of type JobInProgress. In this instantiation, the tasks number of map and reduce is fixed. The purpose JobInProgress persists until the request is satisfied. In an ideal assumption that there are enough resources available across the system to satisfy the request of customers. But in reality, the tasktrackers do not have the same amount of resources. In our case, as reflected in data intensive processing, a new way to

assign tasks to tasktrackers and to manage the sharing of data between mappers and reducers have been implemented.

Here, the need to manager these workflows can be felt and the key is to choose the implementation of an effective strategy.

IV. SCHEDULING PROTOCOL

Performance of any distributed system depends largely on the strategy used to dispatch the various tasks. Thus the main goal of the scheduler is to share resources for several jobs while respecting fairness between these jobs and performance. In this work, we present a scheduling policy that takes into account OLAM MapReduce Tasks Pipelines and Hadoop cluster that execute these tasks.

A. Problem Formulation

The scheduling of MapReduce Pipeline on different nodes that make up a Hadoop cluster is not easy.

A Hadoop job is dispatched by the Master Node (Job Tracker) on various Task Nodes [21]. For our case the problem is more complex because a Task OLAM is a combination of MapReduce Operators, which means that to meet such as request, the system must combine the result of several Hadoop Job. A Job Hadoop will not be considered an action part, to share data between similar applications via execution pipelines.

In fact, many users run the requests on the OLAM system. Each user can launch one or more requests. Give a case as example to capture this complexity in Fig. 3. Users numbered 1, 2 and 3 respectively are launching three, two and one OLAM requests. Each application is, for instance, decomposed into a sequence of three Hadoop job and each node has 18 Mappers. This whole system is not static and that there is an efficient scheduling, we consider competition Mappers on each node and overall priority of requests seen only one user can launch several tasks.

B. Our Scheduling Policy

Our solution solves these problems by introducing a set of simple rules to apply. We publish a rule to apply to each node. Each rule to apply only a limited vision of the problem. Globally, the proposed reactive control is based on the principle of client / server (client / supplier), which is showed in Fig. 4.

And finally the customers of this architecture are queries to be processed, before they undergo various transformations, conducted themselves in the process baths. To do this, they perform two types of treatment: operation to divide these treatments, conducted with mappers, and to unite those treatments, carried out among reduces.

Here, the rules correspond to the decision mechanisms used by the Master Node, to prioritize the various tasks. For our scheduling solution, which is reactive, then we have rules that are related to the general state operations Hadoop cluster. The status of these operations is shown by a set of parameters. For example, we can cite the number of requests per user, the number of nodes and the number of tasks on each node.

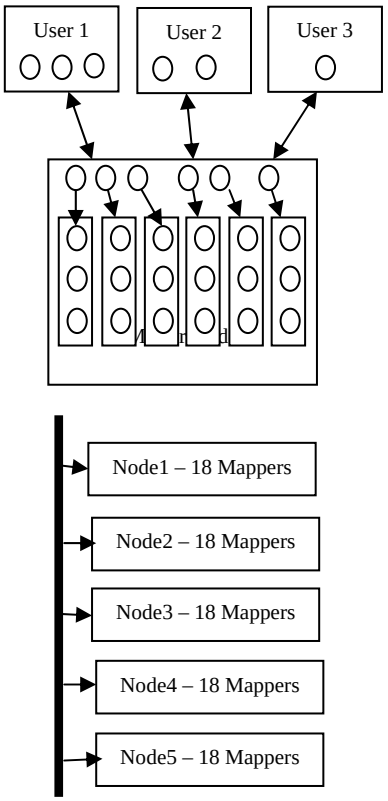

Figure3. Complexity of MapReduce Workflows

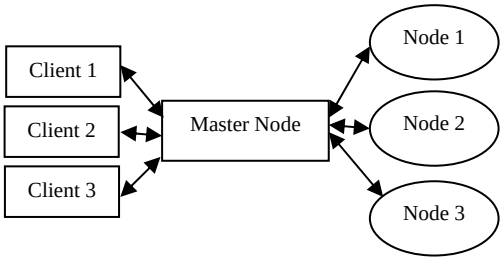

Figure4. Reactive Control based on the Principe of Client/Supplier

Rules

Rule 1: An index is provided at each Job Hadoop. It is directly related to the number of outstanding requests by a user launches.

Rule 2: Each Mapper has an index that combines the Hadoop Job index in which the Mapper is and the arrival number on the node.

Rule 3: Before starting an operation, we must see if there is no need to use intermediate data of similar operations.

C. Scheduling Implementation

In this implementation system, we have a set of rules that takes care of the priority of jobs and cooperation between the various tasks required. Besides that, we also have a state model of the cluster in Fig. 5, which is based on discrete events.

A discrete event model is used to maintain an internal image of the current state of the activity system (Hadoop cluster), and this state evolves according to event occurrences. It usually indicates at any moment all possible decisions, that is to say compatible with the constraints of the system, given a state.

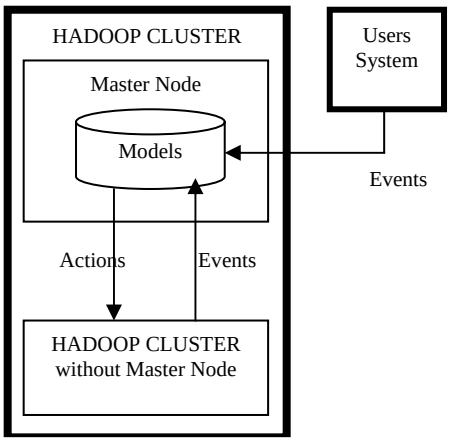

Figure5. Reactive Control Implemented in Hadoop Ecosystem

When the discrete event model indicates that several competing decisions are possible at any given time, it is necessary to make choices; they condition the final performance of the scheduling. These choices are generally produced using priority rules.

We use a simulation tool called MRSIM to experiment. MRSIM is a simulator based on the events that mimic the operation of Hadoop. MRSIM use another simulator appointed GridSim[22], which simulates the behaviour of resources and applications scheduling in parallel and distributed computing. It is also based on JavaSim[23], which is a discrete event simulator. Javasim is an object oriented simulator, which simulates the behaviour of classes. MRSIM allows us to configure a cluster with different racks and characteristics of jobs to run on the cluster. All these files are stored in JSON (JavaScript Object Notation).

Our simulations were conducted with three users who start each request on the simulator; each request can be made from one to several jobs. In turn, a job is characterized by a set of Mappers functions and Reducers functions.

The result of our experience is condensed in table I. We have four jobs submissions on the cluster. We first isolated each job and each of the three users submitted its turn a job. Finally, we simulated the performance of the all three jobs.

By analyzing these data, we see the relevance of our policy as it is the reduction of the execution time of these jobs. All of the simultaneous execution of these jobs also allows us to identify the cumulative force of our police. We interpreted the data differently, trying to identify the gain in percentage of each job submission and the average performance of a job for two cases, namely Running without Policy and Running with policy in Table II.

TABLE I.
SIMULATION RESULTS

	Hadoop Job	Running without policy	Running with policy
User 1	3	40 s	26 s
User 2	2	26.66 s	24.09 s
User 3	1	15.43 s	8 s
All User	6	127.33 s	87.29 s

TABLE II.
SIMULATION RESULTS

	Gain	Job execution average without policy	Job execution average with policy
User 1	35%	13.33 s	8.66 s
User 2	9.6399%	13.33s	12.045 s
User 3	48.15%	15.43 s	8 s
All User	31.44%	21.22 s	15.54 s

The different percentages in Table II highlight the gain in execution time of our strategy. By analyzing the various averages of execution of a job, we find that our environment is subject of disturbances. These disturbances are not related to our scheduling strategy because they are present even before its application. With these disruptions related to the nature of the environment and jobs, we can say that our strategy is well suited for working in this situation.

V. DISCUSSION

We find that our reactive scheduling has the advantage of build real-time and very flexible solution feasible. Types of uncertainty that this method can take into account are very diverse: the heterogeneity of resources (processors and networks), the dynamic and elasticity resources (processors may disappear / (re) appear at any moment), and the performance constraints (minimum execution time), shared among multiple users, management and transfer of large volumes of data, scaling applications and infrastructure, security, etc.. However, we constant that our approach is not intelligent as it is not possible to anticipate the fact that no one knows the schedule once it realized.

To overcome this problem, we believe integrated into the decision phase information some techniques like stochastic approaches where decisions are taken at times stochastic, depending on the past and a priori knowledge of the probability distributions associated with the parameters of activities. We can also draw inspiration from the approaches of artificial intelligence like neural networks and systems of cooperative multi-agents. It is to propose knowledge models of self-organization that provides relevant parameters for the selection rules

VI. Related Work

Scheduling in Hadoop Framework uses the default First In First Out (FIFO) technique [24, 25]. The design and the implementation of this technique are simple, but it suffers from limitations when the environment becomes heterogeneous [25] or specific applications want to run on the platform [26]. JobQueueTaskScheduler and LimitTaskPerJobTaskScheduler are improvements of FIFO scheduler.

JobQueueTaskerScheduler will assign tasks according to a policy priority. In a heartbeat from a TaskTracker, the method assignTasks of scheduler will be called to determine how many tasks will be assigned to the TaskTracker to the origin of the heartbeat. A series of calculations will be performed to calculate the number of free slots on the TaskTracker, so how many tasks he will be able to assign. If the TaskTracker has free slots, you will be able to assign spots map (provided free slots). This is a list of tasks to be run which will be transmitted via the TaskTracker heartbeat response. Same calculations for Reduces a difference: more than a spot of Reduce can be assigned.

LimitTaskPerJobTaskScheduler limit the number of tasks going for a job. It works like JobQueueTaskScheduler but huge differences to note:

- It is possible that several tasks of REDUCE be referred to a TaskTracker (in response to even heartbeat).
- IIt does not include eventual breakdowns tasktrackers: there is no update of the capacity of the cluster.

Must never exceed the number of tasks per job and the burden of taskTrackers.

Hadoop was designed primarily to run large batch jobs such as web indexing and log mining .Users submitted jobs to a queue, and Hadoop ran then in order.

The need to make turn and organize the other types of applications made that Hadoop allows today, to use the scheduler as a plugin component. And now, every one can build his own scheduler according to its needs. It is in direction that Facebook and Yahoo have developed respectively Fair Scheduler and Capacity Scheduler.

In the fair scheduler [11], the jobs share, on average, the resources of the cluster of a way equitable and over time. When there is a single job, this job uses the capacity of the entire cluster. Free slots in pools may be allocated to other pools, while excess capacity within a pool is shared among jobs. And for the capacity Scheduler [11], queues are created and will be shared by the jobs. These queues share the capacity of the cluster. Recent efforts [24] of some research bring improvements such as Dynamic Priority Scheduling [27], Delay Scheduler [28], Resource Aware Scheduling [29] and Deadline Constraint Scheduler [30]. All of these methods allow the Hadoop administrator to adjust the priority assigned to the jobs. However, there is no guarantee that the job will respect a specific deadline, and this issue is handled by some of these schedulers like Deadline Constraint Scheduler.

There are the other types of available schedulers or still outstanding discounted bills of research and development. We can quote some patches [11] from Apache Foundation.

Many attempts to resolve the scheduling problem are as an optimization problem [31] or performance [32].Thus, for example, that Kamal Kc and Kemafor Anyanwu [30] take the deadline as a constraint.

And who said scheduling said complex treatments, said sequences of MapReduce passes, either sequentially or in parallel. The higher the number of passes, the higher it will be difficult to create, but also to keep your data processing. And this is why solutions like Pig [33] and Hive [34] have emerged. Both solutions have some differences but their common point is to provide a new language to express more easily queries. Complex treatments will require going back and forth between these two environments. In short, it is not a pleasant situation.

We can also report some management tools for MapReduce workflows. These tools are very general and do not fit in specific workflows as in our case. The two main are Azkban and Oozie. Azkban[35] is an open source, simple batch scheduler developed at LinkedIn and also Oozie[36] is an open source, services coordinator for producing Hadoop environments developed at Yahoo. Both systems define workflows as DAG (Direct Acyclic Graph) and jobs are defined as java properties in Azkaban and actions in Oozie. There are also Cascading, which is another approach to reducing the complexity of MapReduce workflows. Cascading [37] offers a new Java API on top of the MapReduce layer, allowing developers to be more productive without explicitly by gymnastics of the cutting a treatment in Map / Reduce. Cascading is oriented query processing and data models in the form of a list of named values. Workflows are described as a source taps to collector taps by pipes. It proposes to optimize performance by reducing the number of mapreduce exchanges to minimize the solicitation of hard drives and network.

In Hadoop environments in production, there must be one workflow management tools for synchronizing different components in progress. Failing to choose these tools, there are languages like Python, JRuby, Groovy and help build scripts to coordinate MapReduces job. The performance of a complex distributed system like a Hadoop cluster is conditioned by many parameters, and thus a good scheduling is a compromise between multiple possible decisions. All of these tools require some technical skill and are quite complex for a simple user, and this is one of our goal to present a simple system to use.

VII. Conclusion

In this work, we have proposed models to answer the question of OLAM tasks scheduling on a Hadoop cluster. These models are divided into two parts: the strategy of OLAM MapReduce Operators and a Reactive Scheduling Strategy. Different operators work hand in hand to provide an adequate response to a request from a user. The logic here is that outputs of certain operators are

inputs for others. Our scheduling strategy handles these various flows by managing events emitted by different actors of the system. Our simulation results convince us that a reactive scheduling strategy for Hadoop cluster coordination is an extremely feasible solution to reduce the execution time in an OLAM System.

Future work includes further optimization of our approach and we are going to implement this system, and make tests and analysis on a physical cluster with several machines. We also believe that combining our strategy with another strategy of predictive type may lead to a marked improvement; it is why we also plan to point our future research in this direction. Another research challenge, it will be to gain real time response of the system through Pre-Aggregation. This is not an easy thing as it must choose between the data querying in real-time or uses the preliminary results of aggregation queries. And view the data is huge, the first solution costing in terms of time and the second in terms of space. Therefore, the solution is to find tradeoffs between these two choices.

ACKNOWLEDGMENT

This work was supported in part by the Hunan Provincial Science and Technology Program (Key Program) under Grant No. 2010GK2003.

REFERENCES

[1] J. Han, S. H.S.Chee and J.Y.Chiang, “Issues for On-Line Analytical Mining of Data Warehouses”, SIGMOD'98 Workshop on Research Issues, In *Proc. 1998 SIGMOD’96 Workshop on Research Issues on Data Mining and Knowledge Discovery* (DMKD’98), Seattle: ACM Press, Washington, June 1998, pp. 1-2:5.

[2] M. Usman, S. Asghar, “An Architecture for Integrated Online Analytical Mining”, *Journal of Emerging Technologies in Web Intelligence*, Vol. 3, No. 2, pp. 74-99, 2011.

[3] Bill Immon,"Building the Data Warehouse",John Wiley and Son, 1996

[4] Torben Bach Pedersen , Christian S. Jensen, Multidimensional Database Technology, Computer, v.34 n.12, p.40-46, December 2001

[5] Jens Lechtenborger, Gottfried Vossen. Multidimensional normal forms for data warehouse design. Information Systems 28 (2003) 415–434.

[6] J. Han Y. Fu. “Mining Multiple-Level Association Rules in Large Databases”, *IEEE Transaction On Knowledge And Data Engineering*, Vol. 11, No. 5, pp.798-805, SEPTEMBER/ OCTOBER 1999.

[7] V. Markl, F. Ramsak, R. Bayer, “Improving OLAP Performance by Multidimensional Hierarchical Clustering,” In *Proceedings of the 1999 International Symposium on Database Engineering& Applications*, pp. 165-178.

[8] M. Usman, S. Asghar, S. Fong, “A Conceptual Model for Combining Enhanced OLAP and Data Mining Systems,” In *Proc. of Fifth International Joint Conference on INC, IMS and IDC*,2009, pp. 1958-1963.

[9] S. Chaudhuri, U. Dayal. “An Overview of Data Warehousing and OLAP Technology”, *ACM SIGMOD Record*, Vol. 26, Issue 1, pp. 65-74, 1997.

[10] T. Thalhammer, M. Schref, M. Mohania, “Active Data Warehouses: Complementing OLAP with Analysis Rules”, *Data & Knowledge Engineering*, Vol. 39, Issue 3, pp. 241-269, 2001.

[11] http://hadoop.apache.org.

[12] T. White, “Hadoop: The Definitive Guide”, *O'Reilly Media, Yahoo! Press*, June 5, 2009.

[13] K. Shvachko, H. Kuang, S. Radia, and et al., “The Hadoop Distributed File System”, In *Proc. IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, April 2010, pp. 35-40.

[14] J. Dean, S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters”, *Communications of the ACM*, Vol. 51 Issue 1, January 2008, pp. 107-113.

[15] R.Ben Messaoud, O. Boussaid, and S.L Rabaseda, “A Data Mining-based OLAP Aggregation of Complex Data: Application on XML Documents”, *International Journal of Data Warehousing and Mining*, Vol. 2, No. 4, pp. 1-26, 2006.

[16] J. Han, “OLAP Mining: “An Integration of OLAP with Data Mining”, In *Proceedings of the 7th IFIP Conference on Data Semantics*, Leysin, Switzerland, October 1997, pp. 3-20.

[17] R. Ramakrishnan, B.-C. Chen, “Exploratory Mining in Cube Space”, *Data Mining and Knowledge Discovery*, Vol. 15, No. 1, pp. 29-54, 2007.

[18] S. Sarawagi, R. Agrawal, and N. Megiddo,“Discovery-driven Exploration of OLAP Data Cubes”, In *Proceedings of the 6th International Conference on Extending Database Technology* (EDBT'98), Valencia, Spain, Mars 1998, pp.168-182.

[19] W. Lin, M. Tseng, and M. Wang, “OLAM Cube Selection in On-Line Multidimensional Association Rules Mining System”, *Knowledge-Based Intelligent Information and Engineering Systems*, Volume 3214 , pp. 1276-1282, 2004.

[20] J. Gray, A. Bosworth, A. Layman, and H. Pirahesh, “Datacube: A relational aggregation operator generalizing group by, cross-tab, and sub-total”, In Proc. of *ICDE* 1996, pp. 152–159.

[21] http://wiki.apache.org/hadoop,2012.

[22] Buyya and M. Murshed, “Gridsim: A Toolkit for Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing,” Concurrency and Computation: Practice and Experience, vol. 14, no. 13-15, 2002.

[23] F. Howell and R. McNab, “SimJava: A Discrete Event Simulation Library for Java,” Simulation Series, vol. 30, 1998.

[24] B.Thirumala Rao, B.Thirumala Rao, “Survey on Improved Scheduling in Hadoop MapReduce in Cloud Environments”, *International Journal of Computer Applications* (0975 – 8887) Vol. 34, No.9, pp.29-33, November 2011.

[25] M. Zaharia, A. Konwinski, A. D. Joseph, R. H. Katz, and I. Stoica, “Improving MapReduce Performance in Heterogeneous Environments”,Iin *Proc. of USENIX OSDI*, 2008, pp.29-42.

[26] Z. Ma, L. Gu, “The Limitation of MapReduce: A Probing Case and a Lightweight Solution”, In Proc. of *Conf. on Cloud Computing, GRIDs, and Virtualization* (CLOUD COMPUTING 2010). Nov. 21-26, 2010, Lisbon, Portugal, pp. 68-73.

[27] Thomas Sandholm and Kevin Lai. Dynamic proportional share scheduling in hadoop. In JSSPP ’10: 15th Workshop on Job Scheduling Strategies for Parallel Processing, April,2010

[28] Matei Zaharia, Dhruba Borthakur, Joydeep Sen Sarma, Khaled Elmeleegy, Scott Shenker, and Ion Stoica. Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling. In EuroSys '10: Proceedings of the 5th European conference on Computer systems, pages 265–278, New York, NY, USA, 2010. ACM.
[29] Mark Yong, Nitin Garegrat, Shiwali Mohan: "Towards a Resource Aware Scheduler in Hadoop" in Proc. ICWS, 2009, pp:102-109
[30] K. Kc, K. Anyanwu, "Scheduling Hadoop Jobs to Meet Deadlines", In *Proceedings of IEEE International Conference on Cloud Computing Technology and Science (Cloudcom)*, Indianapolis, USA, Nov, 2010, pp. 388-392.
[31] H. Chang, M. Kodialam, R. Kompella, T. V. Lakshman, M. Lee, S. Mukherjee, "Scheduling in MapReduce-like Systems for Fast Completion Time", In *Proceedings of IEEE INFOCOM*, 2011, pp. 3074-3082.
[32] J. Polo, D. Carrera, Y. Becerra, and J. Torres, "Performance-Driven Task Co-Scheduling for MapReduce Environments", In *Proceedings of 12th IEEE/IFIP Network Operations and Management Symposium* IEEE, 2010, pp.373-380.
[33] http://pig.apache.org/,2012.
[34] http://hive.apache.org/,2012.
[35] http://data.linkedin.com/opensource/azkaban,2012.
[36] http://incubator.apache.org/oozie/,2012.
[37] http://www.cascading.org/about/,2012.

Shengbing Ren was born on August 4, 1969 at Yueyang City, Hunan Province, China. He received his Bachelor of Science degree in Computer Software from Department of Mathematics, Huazhong Normal University, Hubei Province, China in 1992. He received his Master's degree in Computer Application Technology in 1995 from Department of Computer, Central South University of Technology, Hunan Province, China. He received his Doctor's degree in Control Theory and Control Engineering in 2007 from School of Information Science and Engineering, Central South University, Hunan Province, China. His research interests include: software engineering, embedded system, network on chip, image processing.

He is an associated professor in School of Software, Central South University, China. He is a Senior Member of China Computer Federation. He accomplished 10 research projects, including 2 the National Natural Science Foundation of China as a key member, and over 40 papers was published. Now, He is dedicated to the research concentrated mostly on software evolution and hardware software co-design.

Dieudonne Muheto was born in Ngozi, Burundi, on October 13, 1984 and received the BS. degree in Computer Science from Greats Lakes University, Burundi, in December 2009. From 2009 to 2011, he was a software developer of Soft Center Burundi. Since September 2011, he is M.S candidate in Central South University.

His research interests include distributed systems and data analytics.