Secure Max-Min Fair Resource Allocation for a Selective Field Of Stream Big Data Analytics in Shared Cloud

Nivethini Manickavasagam[1], **Deepa Kuppusamy**[2], **Prathipa Baskaran**[3], **Veeralakshmi Ponnuramu**[4]

[1] *Student, IT, Prince shri venkateshwara padmavathy engineering college, TN, India*

[2] *Student, IT, Prince shri venkateshwara padmavathy engineering college, TN, India*

[3] *Assistant professor, IT, Prince shri venkateshwara padmavathy engineering college, TN, India*

[4] *Associate professor, IT, Prince shri venkateshwara padmavathy engineering college, TN, India*

ABSTRACT

In a distributed environment, there is some complexity for resource allocation. The problem resource allocation faces are complication on time, space and data volumes. The dynamic incoming data are uploaded by multiple user login through internet. In stream big data analytics, the data processing workflow is abstracted as directed graph referred as a topology. Data are read from storage and processed tuple by tuple, because the content of data is in the format of meta store. The performance of topology is evaluated by its throughput. This paper proposes an efficient resource allocation scheme for a heterogeneous shared stream big data analytics cluster shared by multiple topologies by using Max-Min Resource Allocation (MFRAX) algorithm. To maintain the security in the cloud, the data will be converted into secure format by using K-anonymity algorithm. The combination of MFRAX and K-anonymity algorithm enrich the resource allocation and security.

Keywords*:* *streaming, topology, resource allocation, security.*

1.INTRODUCTION

Big data is a term that describes the large volume of data, both structured and unstructured. The data analytics is usually performed on a distributed environment. Analytics can be classified into two different approaches: in batches and in streams. Batch processing is the execution of a series of jobs in a program on a computer without manual intervention. Dataset is divided into multiple smaller subsets to be processed in parallel. The analytics result for the dataset is returned after the processing results of the subsets are summarized. Apart from batch processing stream processing were used to tackle the unbounded data which are generated, gathered, and stored continuously. The dynamic incoming data are processed sequentially and incrementally on a record by record basis. One typical example of such data is the tweets of twitter, which are generated at a rate of around 350000 tweets per minute (as of year 2016). An important characteristic of stream processing is to compute the results corresponding to dynamic incoming data. For example search the recent topic in the twitter which are analytic and the results in the twitter data are updated dynamically in the twitter homepage. Hadoop is an open-source software framework for storing data and running applications on cluster of commodity hardware. The main advantage of Hadoop is ability to store and process huge amounts of any kind of data quickly. Hadoop distributed file system (HDFS) is used to compute the data in high speed and retrieve the data in an efficient way. In cloud computing environment the fair resource allocation policy implemented in Yet Another Resource Negotiator (YARN), which provides resource management for the processes running on Hadoop. Flume is a software that collects, aggregates and moves large amounts of streaming data into HDFS. MapReduce is a parallel processing software framework. It is comprised of two steps. Map step is a master node that takes inputs and partitions them into smaller sub problems and then distributes them to worker nodes. After the map step has taken place, the master node takes the answers to all of the sub problems and combines them to produce output. Map function performs filtering and sorting operation. Reduce function performs summary operation. HBase is a non relational distributed database that runs on top of Hadoop. HBase tables can serve as input and output for MapReduce jobs. Cloud sharing is a system in which a user is allotted a storage space on a server. To convert the data into secure format we used k-anonymity algorithm which is not understandable to another user in a cloud environment. Necessary background and related work are discussed in section 2. The algorithm design are explain the resource allocation in section 3. Finally, section 4 concludes this paper.

2.RELATED WORK

In streaming big data analytics resource allocation scheme was researched recently.[1]addressed the efficient analysis of streaming big data in heterogeneous collection of computer. Max-min fairness, proposed in 1981 independently by Jaffe[3] and Hayden[4], has been widely adopted metric in measuring the fairness of resource allocation over the flow control in integrated packet networks. In [1], dynamically monitored the resource demands on tasks according to in and out of incoming data. Xu et al. [8] proposed a scheduler which is used to reduce the inter-node traffic. Based on the application scenario described in the aforementioned literature, we propose an efficient resource allocation scheme for stream big data analytics in shared cloud when the data is retrieved from storage area we are applied security algorithm on the data. In proposed system the data are stored in different field using MapReduce function. Resource allocation problem and propose problem-specific allocation algorithms for various applications under different scenario and settings, such as routing and flow allocation in networks [11], [12], cloud computing resource allocation [15] and Hadoop job scheduling [16]. In this paper, we propose efficient and problem-specific allocation algorithm accordingly.

3.PROPOSED WORK

Multiple user have uploaded the data dynamically through the internet. The data is called as streaming data. Resource capacity are monitored and oozie are used to schedule the job. Flume is a software that collects, aggregates and moves large amounts of streaming data into HDFS. Yet Another Resource Negotiator (YARN) provides the resource management for the processes running on Hadoop. HBase is a non- relational, distributed database that runs on top of Hadoop. HBase tables can serve as input and output for MapReduce jobs.

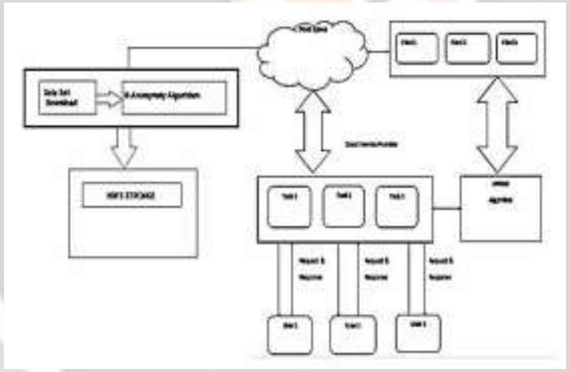

Fig-1:*resource allocation for streaming data and stored procedure.*

Memory analytics are done by Spark. Zookeeper is an application that coordinates distributed processing. All these process are done in front end shown in Figure.1. Communication bus transfer the data between the front end and back end. Discovery of data is also known as data extraction, i.e. the act or process of retrieving data out of data source for further data processing or data storage. We have used MapReduce concept to separate the field of data. The structured and unstructured data are retrieved from cloud storage area. Cloud sharing is a system in which a user is allotted a storage space on a server. MapReduce is a framework for processing large data sets in parallel across a Hadoop cluster. Data analysis uses a two steps map and reduce process. The map phase counts the words in each document, then the reduce phase aggregates the pre-document data into word and counts spanning the entire collection. The mapper job is to process the input data and splitting the data into small chunks. Reduce stage is the combination of both shuffle and reduce. The reducer's job is to process the data that come from the mapper. Name node is a node that manages the Hadoop Distributed File System (HDFS).Data node is a node where data is presented in advance before any processing takes place. Master node and slave node performs the map and reduce operation.

4.ALGORITHM

To solve the problem in resource allocation, MFRAX algorithm to be used. MFRAX is a linearized approach, which is based on the approach of branch and bound of the cluster nodes. The utility function is making a huge problem in the resource allocation. Initial data set is the trained data set which matches the continuous test data set in the environment and allocated it into a multiple topologies after data extraction. It's a heuristic approach which is used to find the optimal solution for resource allocation problem.

Table-1:summary of key notation

I	Topology index
J	Task index
B	Throughput of topology i
	Utility function
	Upper bound of throughput for topology i
	Resource to throughput ratio of task j from topology i
M	Node index
K	Set of node in the cluster
	Resource capacity of node m

```
Algorithm 1 MFRAX solution method
1:    for each node m ∈ k do
2: Do a bi-section search for the targeted objective value which satisfies
   Σ( , )∈    −1( )    =
3: Find set that contain all the saturated topologies on a node m, i.e..,
   = { | − 1( ) > , : ( , ) ∈    }
4: set * = { ( )} if ≠ ∅,
   ∈
   and * = otherwise
```

Algorithm 1 is the proposed solution method for MFRAX problem. Each individual problem is solved (step 1-6), and optimal solution is attained. The basic idea to solve each MFRAX problem is that when optimal condition is achieved on node k, generally each topology that has tasks on that node tends to have the same utility so that no single topology has the minimum utility value to lag the objective value (step 2). The saturated topology is (step 3). Saturated topology is a topology that cannot reach the same targeted utility value as others since the value is out of its utility range. There is increasing pressure to share health information and even make it publicly available. However, such disclosures of personal health information raise serious privacy concern. To alleviate such concern, it is possible to anonymize the data before disclosure. One popular anonymization approach is k-anonymity. There have been no evaluations of the actual re-identification probability of k-anonymized data sets. The way in which k-anonymity would be applied depends on the re- identification scenario one is protecting against. To protect against the prosecutor re- identification scenario, then k-anonymity should be used. If the prosecutor scenario is not applicable, then k-anonymity is not recommended, and k-map should be used instead. If both scenario are plausible, then k-anonymity should be used because this is the most protective. Therefore, being able to make a decision on whether the prosecutor scenario is applicable is important. In this paper we focused on k-anonymity, which is a popular approach for protecting privacy.

5.CONCLUSION

The resource allocation for multiple heterogeneous topologies in a heterogeneous stream big data analytic cluster to achieve max-min fairness in utilities of the topology throughputs has been observed. Our proposed system produce high performance and reduce the computational complexity. The security are applied to the data, when the information is retrieved from the data source. For the future work, least recently used page replacement algorithm used to delete the unused data in the storage area so we can reduce the time complexity of our system. When the data are not user longer lime we used this method. Ranking method are used to view more used data so the multiple user can understand which data can viewed by more user. For more security purpose we will apply security algorithm like AES and DES. These algorithm is more secure than k-anonymization algorithm.

6.REFERENCES

[1] M. Rychly, P. Koda et al., “Scheduling decisions in stream processing on heterogeneous clusters”, in Proc. Int. Conf. Complex,Intelligent and Software Intensive Systems, 2014, pp. 614–619.

[2] J. S. v. d. Veen, B. v. d. Waaij et al., “Dynamically scaling Apache Storm for the analysis of streaming data,” in Proc. IEEE Int. Conf.Big Data Computing Service and Applications, 2015, pp. 154–161.

[3] J. M. Jaffe, “Bottleneck flow control,” IEEE Trans. Commun., vol. 29,no. 7, pp. 954–962, 1981.

[4] H. P. Hayden, “Voice flow control in integrated packet networks,”DTIC Document, Tech. Rep., 1981.

[5] Apache Hadoop. [Online]. Available: http://hadoop.apache.org/

[6] M. Zaharia, M. Chowdhury et al., “Spark: cluster computing with working sets,” in Proc. 2nd USENIX Conf. Hot Topics in Cloud Computing, vol. 10, 2010, p. 10.

[7] New Tweets per second record, and how! [Online]. Available: https://blog.twitter.com/2013/new-tweets-persecond-record-and-how

[8] J. Xu, Z. Chen et al., “T-Storm: traffic- aware online scheduling in Storm,” in Proc. IEEE Int. Conf. Distributed Computing Systems, 2014, pp. 535–544.

[9] Apache Storm. [Online]. Available: http://storm.apache.org/

[10] Y. Jiang, “A survey of task allocation and load balancing in distributed systems,” IEEE Trans. Parallel Distrib. Syst., vol. 27, no. 2, pp. 585–599, 2015.

[11] J. Kleinberg, Y. Rabani et al., “Fairness in routing and load balancing,” in Proc. 40th IEEE Annu. Symp. Foundations of Computer Science, 1999, pp. 568–578.

[12] B. Radunovi´c and J.-Y. L. Boudec, “A unified framework for maxmin and min-max fairness with applications,” IEEE/ACM Trans. Netw., vol. 15, no. 5, pp. 1073–1083, 2007.

[13] JStorm. [Online]. Available: https://github.com/alibaba/jstorm

[14] A. Toshniwal, S. Taneja et al., “Storm@ Twitter,” in Proc. ACM SIGMOD Int. Conf. Management of Data, 2014, pp. 147–156.

[15] D. Zarchy, D. Hay et al., “Capturing resource tradeoffs in fair multi-resource allocation,” in Proc. IEEE Int. Conf. Comput. Commun., 2015, pp. 1062–1070.

[16] Z. Huang, B. Balasubramanian et al., “Need for speed: CORA scheduler for optimizing completion-times in the cloud,” in Proc. IEEE Int. Conf. Computer Communications, 2015, pp. 891–899.

[17] P. Li, S. Guo et al., “Max-min lifetime optimization for cooperative communications in multi-channel wireless networks,” IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 6, pp. 1533–1542, 2014.

[18] Y. Bejerano, S.-J. Han et al., “Fairness and load balancing in wireless lans using association control,” in Proc. 10th ACM Annu. Int. Conf. Mobile Computing and Networking, 2004, pp. 315–329.

[19] C. H. Papadimitriou and K. Steiglitz, Combinatorial optimization: algorithms and complexity. Courier Corporation, 1982.

[20] H. Kellerer, U. Pferschy, and D. Pisinger, Introduction to NPCompleteness of knapsack problems. Springer, 2004