ISSN:0975-9646

Data Mining With Incremental Mapreduce

Trupti Maruti Shinde
Research Scholar,
Department of Computer Engineering,
Dr. D. Y. Patil Institute of Engineering &Technology,
Pimpri, Pune, India.

Prof. Santosh V.Chobe
Associate Professor,
Department of Computer Engineering,
Dr. D. Y. Patil Institute of Engineering &Technology,
Pimpri, Pune, India.

Abstract **- New data and updates are continuously arriving, that's why the results of data mining applications become stale and obsolete over time. It is unable to provide meaningful insights into new decisions due to its stale nature. Incremental processing is an approach towards refreshing those mining results, which uses previously saved states to avoid cost of re-computation from scratch. Incremental MapReduce is an extension to MapReduce, which provides incremental processing for mining one step and iterative data. Incremental MapReduce performs key-value pair level incremental processing somewhat better than task level re-computation which were provided by existing approaches like MapReduce, IncMR, Incoop, and iMapReduce. Incremental MapReduce make use of a one-step algorithm and iterative algorithms with diverse computation which will provide more efficiency than existing approaches which were incur unnecessary re-computation time and cost.**

Keywords- *Data Mining, MapReduce, Incoop, IncMR, MRBGraph.*

I. INTRODUCTION

The vast amount of digital data is being collected in many important areas, like electronic commerce, Social media, finance, health care, education, and environment. Now it is necessity of mining that data to get meaningful insights into the business needs which will assist to take business decisions and also affords to an individual advanced quality services. Computing frameworks are available which are developed for big data analysis, MapReduce is (with its open-source implementations, such as Hadoop), the most generally used in production. The Incremental MapReduce enhances the existing MapReduce with added features. Big data is constantly changing, when new data and updates are being collected, the input data of a big data mining algorithm will gradually change, and the computed results will be outdated as time passes. In some circumstances, it is very important to refresh those mining results periodically, so that the mining results become up-to-date and in time. Data undergoes many changes, as variety of data continuously arrives. So existing data changes, due to this result of data mining applications become stale and obsolete as time passes, so refreshing the mining results to get insights into current updated data is time consuming as well as incurs high cost, as it takes into consideration data processing from scratch. So there is need to do incremental processing, which will save states of previous data and performs mining on only that data, which are affected by changes, which saves time and doesn't incur high cost. Incremental MapReduce, an extension to MapReduce which offers both incremental and iterative processing for both one step and iterative computation.

II. LITERATURE SURVEY

In [2]. J. Dean and S. Ghemawat discussed about MapReduce framework, MapReduce library groups together all intermediate values associated with the same intermediate key and passes them to the Reduce function, and the Reduce function, also written by the user, accepts an intermediate key and a set of values for that key, it merges together these values to form a possibly smaller set of values. MapReduce are automatically parallelized which are executed on large set of machines but it incurs unnecessary I/O overhead.

In [3]. P. Bhatotia , et al. focused on Incoop framework which is an extension to hadoop for incremental processing. Incoop identifies changes in the inputs and allows the automatic update in the outputs by using an efficient, detailed result re-use mechanism. Incoop, allows existing MapReduce program's incremental processing but with efficient modifications in input and with reuse of intermediate results from previous computations but it supports task level incremental processing.

In [4]. M.Zaharia, et al. proposed iMapReduce framework, which enables users to state the iterative operations with map and reduce functions, at the same time supporting the automatic iteration regardless of user participation. iMapReduce extensively improves the performance of iterative algorithms by -

I. Avoiding the creation of new job for each iteration.
II. Enables beginning of new map task regardless of previous reduce tasks completion.
But lack in support for incremental with iterative processing.

In[5].Y.Zhang, et al. put their great insights onto IncMR framework, IncMR is a enhanced model for large-scale incremental data processing ,that uses the original MapReduce model. Without any modification in existing Map-Reduce model IncMR delivers incremental processing. It supports incremental data processing, intermediate state preservation and incremental map and reduce functions. It enables incremental processing with task-level re-computation, but in this framework users themselves have to manipulate the states.

In [6].C. Yan, et.al put their great thoughts on Spark System, which is a distributed memory abstraction which allows programmers to process in-memory computations on large clusters with fault-tolerant capacity. Spark gives better performance than Hadoop due to it's in-memory processing nature, but it's performance suffers if input and intermediate data are not able to fit into memory.
In [10]. Trupti A. Kumbhare, Prof. Santosh V. Chobe focused on association rules which are mostly used in data mining to find patterns based on association relationship for two different data sets and put their insights on market basket analysis.

III. SYSTEM ARCHITECTURE

The system may perform well when the data is either one step or iterative in nature as well as it performs fine grain level incremental processing to avoid task level re-computation.

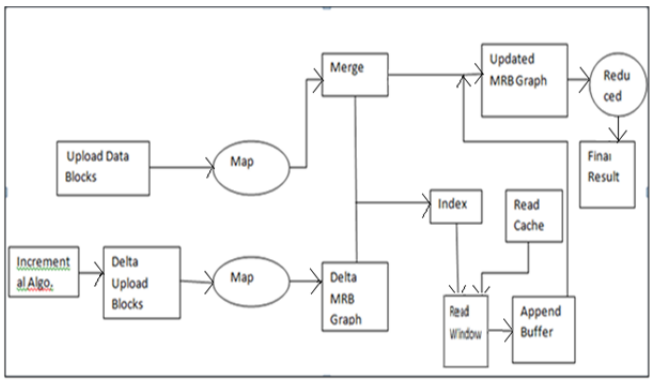

Fig.1 System Architecture

System performs iterative as well as incremental processing for refreshing data mining results to keep them up-to-date.
-In this system firstly data blocks are uploaded on which system will perform incremental iterative processing. Incremental algorithms are applied on delta upload blocks which undergo insert/update/delete operation.
-Final graph will create with updated MRBGraph (Map-Reduce bipartite graph). This updated MRBGraph are appended to an existing uploaded blocks which are not going through Insert/Update/Delete operation or remains same.
-Finally we will get reduced graph which will show all relation between different users from which some are existing users and some are newly added/deleted/updated.

Hypothesis and Mathematics-
Consider two MapReduce jobs A1 and A1′, performing the same computation on input data set D1 and D1′, respectively. D1′=D1+ΔD1, where ΔD1 consists of the inserted and deleted input (K1, V1)s. An update can be represented as a deletion followed by an insertion, re-compute only the Map and Reduce function call instances that are affected by ΔD1. It will simply invoke the Map function for the inserted or deleted (K1,V1)s, since the other input kv-pairs are not changed, their Map computation will remain the same, then computed the delta intermediate values, denoted ΔM,including inserted and deleted (K2; V2)s then will re-compute the Reduce function for each K2 in ΔM. The other K2 in M will not see any changed intermediate values and therefore will generate the same final result. For a K2 in ΔM, typically only a subset of the list of V2 will change. This will retrieve the saved (K2, {V2})s from M, and will apply the inserted and/or deleted values from ΔM to obtain an updated Reduce input. It will re-compute the Reduce function on this input to generate the changed final results (K3,V3)s. Generated results from this incremental computation will tell whether they are the the same as the results from completely re-computing A′ or not.

Relevant Mathematics-

V: state data (updated on every iteration)
D: structure data (static during iterative computation) //graph
F ( ): iterative update function

$v^k = f(v^{k-1}, D)$
R: scores R //ranking interlink (values) linkage, how much links are interlinked

W: web graph matrix
$R^{(k)}$=dW$R^{(k-1)}$+(1-d)E

Utilize the previous iterative computation's result:
-Reduce the number of iterations
• Structure data is slightly changed = the result is slightly changed
• Start from the previously converged state rather than from a random start point

$v^k = f(v^{k-1}, D)$ $\quad v^k = f(v^{k-1}, D + \Delta D)$

-Reduce the workload of each iteration as below-

O ($|D + \Delta D|$) ⇨ O($|\Delta D|$)

Flowchart of system-
Flowchart shows how system will work step-by-step.

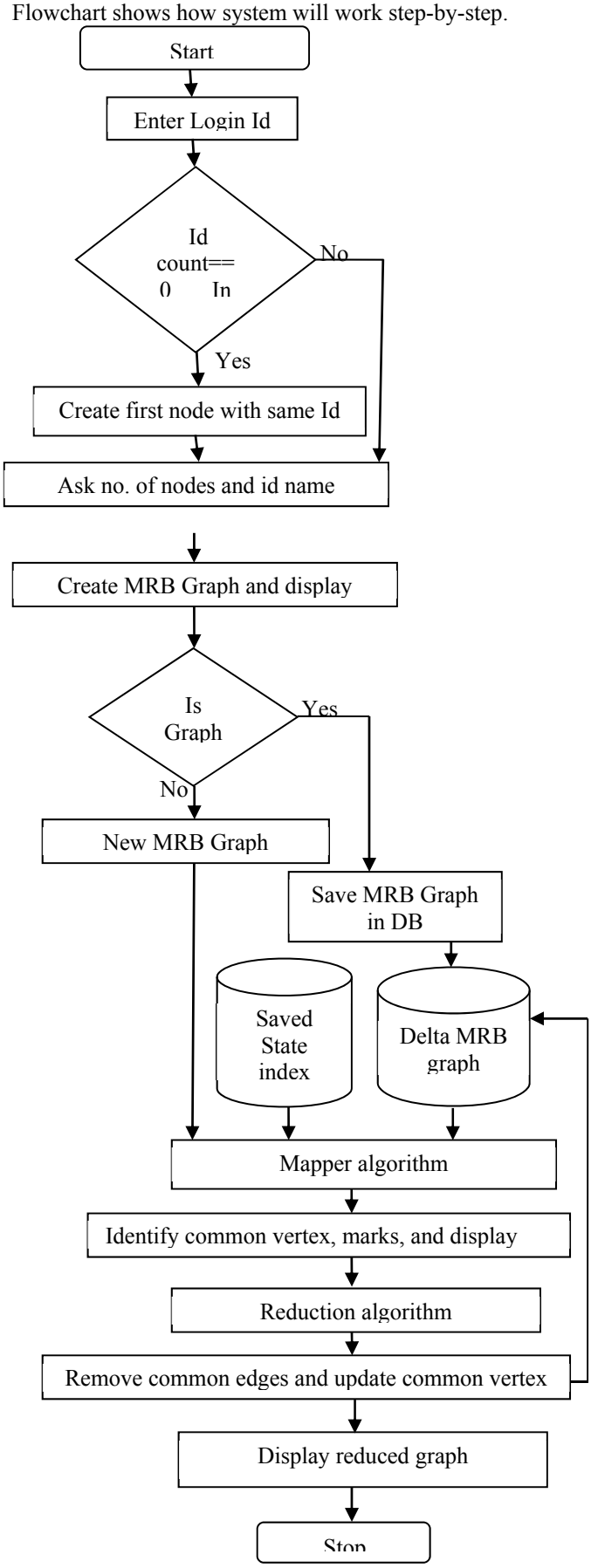

Fig.2 Flowchart of System

Generally the system involves the main functions like creation of graph, saving states of graph, identify changes in the graph and mapping them for identifying common nodes and edges as well as reducing those common nodes and edges to make it simple, and time saving still productive.

-After logging into the system ,it first checks if there is existing graph nodes or not, if no then it will create first node and prompt about how many number of nodes user wants and their respective name. After getting number and name of nodes system will create MRB Graph and displays it. If there are existing nodes then it will prompt for new number of nodes and name of those nodes for creating MRB graph.
-If created graph doesn't have any new interlinks then that graph will be stored in DB.
-But if it finds any new interlinks then it will create new MRB Graph. All states are saved in DB for indexing purpose and for identifying changes in various states.
-Saved MRB Graph will go through test for changes made in graph (newly inserted/updated/deleted) nodes.
- Newly created MRB Graph/ Delta MRB Graph and saved states provides an input to mapper algorithm, which will identify common vertex or nodes, marks them and displays them. Reduction Algorithm reduce common nodes, common edges, and generates reduced graph.

IV. RESULTS

Suppose user sanskar has registered into this system and logged on to system to create his own network as shown in Fig.3.

Fig. 3 Login page

After this he will create his own network with registered users by adding users/friends as shown in fig. 4 .

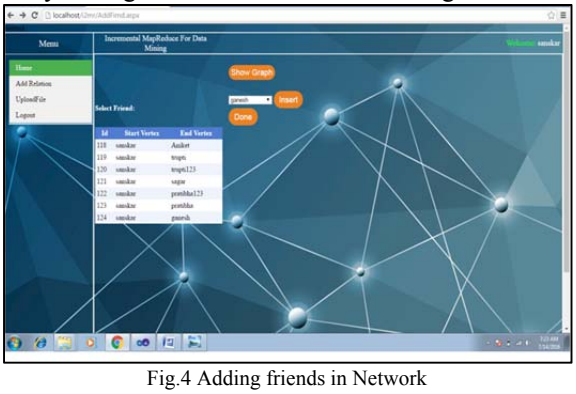

Fig.4 Adding friends in Network

The resultant graph is as shown in fig. 5

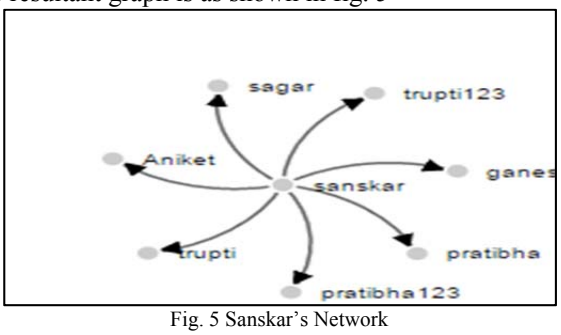

Fig. 5 Sanskar's Network

Admin can view the relationships amongst all registered users, which is an optimized and merged graph. This graph combines registered user's network into one single graph, with an incremental processing rather than processing from scratch to save re-computation cost and time.

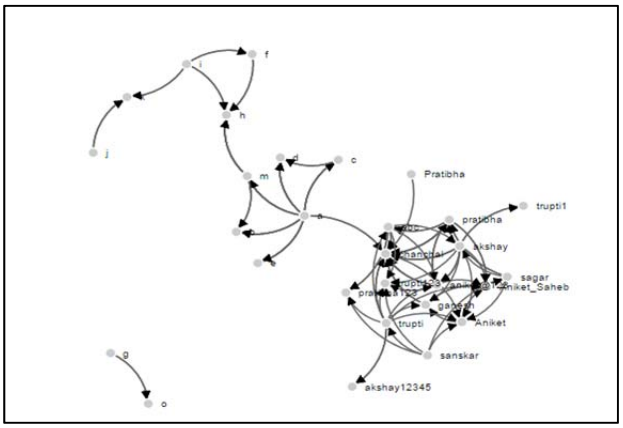

Fig.6 Final All User's Network

System starts from new user and find all his /her friends to create network. finally all registered user's network come in to picture with all possible relationships as shown in Fig 6.

Below mentioned timing diagram shows how results of this system will be reducing re-computation time for iterative and incremental processing.

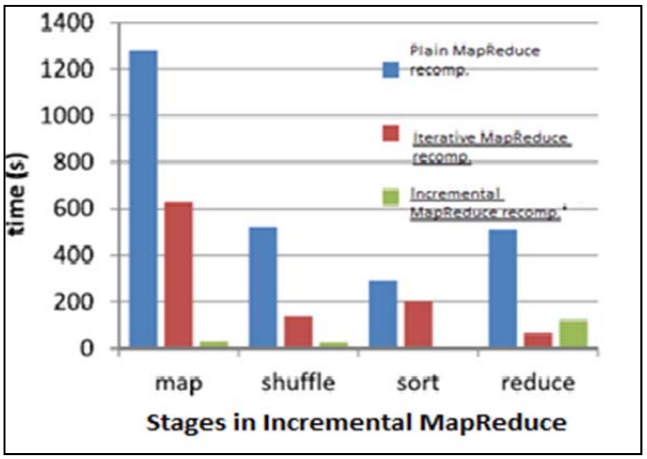

Fig.6 Timing Diagram

CONCLUSION

Data mining provides meaningful insights into vast dataset which always undergoes changes, so there is need to refresh that data continuously to keep it up-to-date and mines currently refreshed and up-to-date data. Existing approaches creates unnecessary re-computation while data mining with incremental mapreduce addresses the problem of refreshing mining results, which consist of one step and iterative as well as incremental dataset. This approach will provide the solution to the problem of refreshing mining results of incremental one step and iterative dataset, within short time and less cost with reduction in unnecessary re-computation.

.

REFERENCES

[1] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," Proceedings. 6th Conference.Symposium Opear System.Des Implementation, p. 10, 2004.

[2] P. Bhatotia, A. Wieder, R. Rodrigues, U. A. Acar, and R. Pasquin,"Incoop: Mapreduce for incremental computations,"Proceedings, 2nd ACM Symposium Cloud Computing, pp. 7:1–7:14, 2011.

[3] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for, in-memory cluster computing," in Proceedings 9th USENIX Conference Network System Des. Implementation, p. 2,2012.

[4] Y. Zhang, Q. Gao, L. Gao, and C. Wang, "imapreduce: A distributed computing framework for iterative computation," J. Grid Computing., volume. 10, no. 1, pp. 47–68, 2012.

[6] C. Yan, X. Yang, Z. Yu, M. Li, and X. Li, "IncMR: Incremental data processing based on mapreduce," in Proceedings, IEEE 5th International Conference. Cloud Computing., pp 534–541, 2012.

[7] T. Jeorg, R. Parvizi, H. Yong, and S. Dessloch, Incremental re-computations in mapreduce, in Proceedings. 3rd International Workshop Cloud Data Manage, 2011.

[8] G. Malewicz, M. H. Austern, A. J. Bik, J. C. Dehnert, I. Horn,N. Leiser, and G.Czajkowski, Pregel: A system for large-scale graph processing, in Proceedings. ACMSIGMOD International Conference Manage Data, 2010.

[9] Y. Bu, V. Borkar, J. Jia, M. J. Carey, and T. Condie, Pregelix: Big(ger) graph analytics on a dataflow engine, in Proceedings VLDB Endowmen,2015.

[10] Trupti A. Kumbhare, Prof. Santosh V. Chobe, An Overview of Association Rule Mining Algorithms, in International Journal of Computer Science and Information Technologies, Vol. 5 (1) , 2014, 927-930 .