Research Journal of Applied Sciences, Engineering and Technology 8(2): 160-166, 2014
DOI:10.19026/rjaset.8.955
ISSN: 2040-7459; e-ISSN: 2040-7467
© 2014 Maxwell Scientific Publication Corp.
Submitted: January 20, 2014 Accepted: February 15, 2014 Published: July 10, 2014

Research Article

Apriori Association Rule Algorithms using VMware Environment

[1]R. Sumithra, [2]Sujni Paul and [3]D. Ponmary Pushpa Latha
[1]School of Computer Science, CMS College, Coimbatore, India
[2]Department of MCA, OXFORD College of Engineering, Bangalore, India
[3]Department of Computer Applications, Karunya University, Coimbatore, India

Abstract: The aim of this study is to carry out a research in distributed data mining using cloud platform. Distributed Data mining becomes a vital component of big data analytics due to the development of network and distributed technology. Map-reduce hadoop framework is a very familiar concept in big data analytics. Association rule algorithm is one of the popular data mining techniques which finds the relationships between different transactions. A work has been executed using weighted apriori and hash T apriori algorithms for association rule mining on a map reduce hadoop framework using a retail data set of transactions. This study describes the above concepts, explains the experiment carried out with retail data set on a VMW are environment and compares the performances of weighted apriori and hash-T apriori algorithms in terms of memory and time.

Keywords: Association rules, cloud mining, hadoop, hash-T, map-reduce, W-apriori

INTRODUCTION

Knowledge discovery in databases also called as data mining can extract useful hidden information from the datasets which are massive, noise and vague. Association rule, one of the important techniques can find out the relationship between item sets in the database of transactions. Previous association rule mining algorithms (Agrawal and Srikant, 1994) faces many difficulties such as time constraints. Apriori uses an iterative approach which incurs high I/O overhead for scanning. Parallel association rules (Paul and Saravanan, 2008) have come up to speed up the process with their own weaknesses of communication and synchronization. Now map-reduce model with its own key benefits of handling failures and hiding complexity of fault-tolerance gives a distributed environment to improve the apriori algorithm. This study contributes to distributed data mining by:

- Using weighted apriori and Hash T apriori for the mining process
- By using map-reduce model which splits a large problem space into small pieces and automatically parallelizes the execution of small tasks on the smaller space for each algorithm
- Using apache's map-reduce implementation-Hadoop Distributed File System (HDFS) for the dataset to compare the weighted apriori and hash T apriori using different nodes

LITERATURE REVIEW

Traditional apriori: The support and confidence are common indicators to measure strength of association rules. Minimum thresholds are better constraints on support and confidence. Apriori is applied to count the support of itemsets with a breadth-first search strategy and generates candidate itemsets.

Weighted apriori: Classical model of association rule mining uses the support measure in which every transaction is treated equally. Actually different transactions have different weights in real life data sets. WARM (Sun and Bai, 2008) introduces w-support a new measure which is a link based measure, which calculates the authority weight, auth (i) represents the significance of an item i. w-suppoirt can be regarded as a generalization of support, which takes the weights of transactions into account. These are calculated using a global link structure of the database. W-support is more better than counting based measurement.

Hash-T apriori: Hash-T algorithm (Grudziński and Wojciechowski, 2009) overcomes the weaknesses of the apriori algorithm by reducing the number of candidate k-itemsets. In particular 2-itemsets is the key to improve the performance. The number of itemsets in c2 can be made less using hashing techniques so that scan required to determine L2 is more efficient. A hash function hash (i.e.,) maps them into the different buckets of a hash Table 1 structure and increases bucket

Corresponding Author: R. Sumithra, School of Computer Science, CMS College, Coimbatore, India

Table 1: Association rules generated in each algorithm

Iterations	W-apriori	Hash-T
1	354	354
2	237	152
3	102	75
4	19	17
5	1	1
Total	713 (rules)	599 (rules)

counts. A 2-itemset whose corresponding bucket count in the hash Table 1 is below the support threshold cannot be frequent and thus should be removed from the candidate set.

METHODOLOGY

Map-reduce: Map reduce uses data parallel model. It is a patented software framework (Yang *et al*., 2010) introduced by Google to support distributed computing on large data sets on cluster of computers. It performs the computations as two functions, Map and Reduce. Map reduce provides an abstraction that involves the programmer defining a 'mapper' and a 'reducer' with the following signatures:

Map: (key 1) => list (key 2, value 2)
Reduce: (key 2, list (value 2) =>list (value 2)

Hadoop: Hadoop is popular open source implementations of map reduce. This tool is designed for analysis and transformation of very large data sets. It uses a Hadoop Distributed File System (HDFS). HDFS (Kambatla *et al*., 2009) schedules map and reduce tasks to distributed resources which handles many tough problems including parallelization, concurrency control, network communication and fault tolerance.

Map reduce for apriori:
Traditional apriori: (Li *et al*., 2012)
Input: D (Database of transactions)
Min_sup: (Minimum support threshold)
Output: L (Frequent itemset)

Data can very well be distributed in multiple nodes using map-reduce hadoop platform and apriori algorithm can be applied. Minimum support and confidence are to be taken as specified above. Figure 1 and Table 2 explains all these concepts.

Figure 1 explains the map-reduce model used to implement the apriori algorithm. M specifies the Map model which is applying the algorithm to the individual split data and Reduce model is consolidating the resulting association rules from all the nodes.

Map reduce using weighted apriori and hash T algorithms: Based upon the above traditional apriori map-reduce model a new map-reduce model for weighted apriori and hash-t apriori algorithms is designed. Figure 2 describes the model in which 'task'

Table 2: Key/value pairs for map function and reduce function

Input/output	Map function	Reduce function
Input: key/value pairs	Key: line No.; value: one row of data	Key: candidate item sets; value: 1
Output: key/value pairs	Key: candidate itemsets; value: 1	Key: frequent subitems; value: support

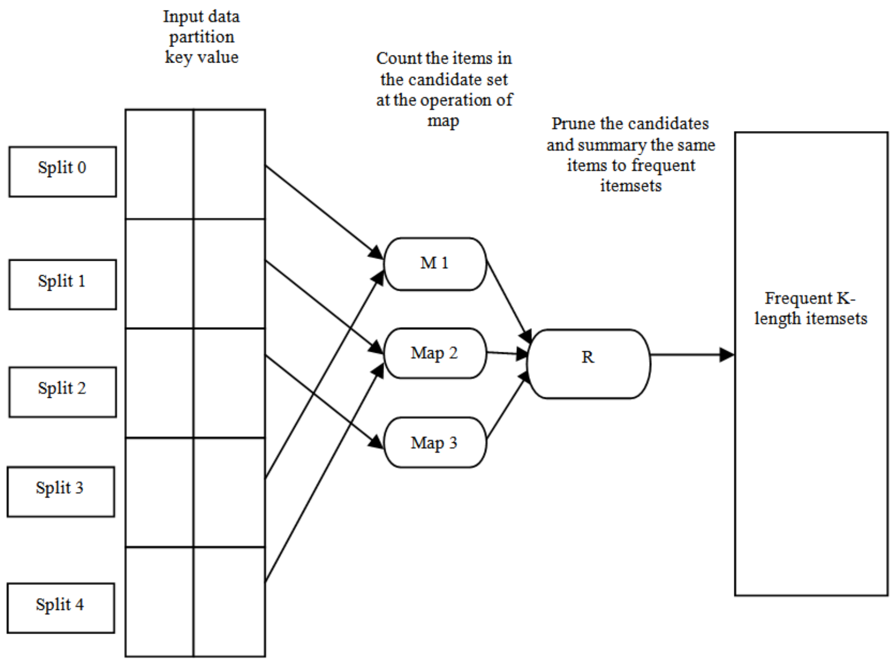

Input data partition key value
Count the items in the candidate set at the operation of map
Prune the candidates and summary the same items to frequent itemsets
Split 0
Split 1
Split 2
Split 3
Split 4
M 1
Map 2
Map 3
R
Frequent K-length itemsets

Fig. 1: Map reduce model for apriori (M-map, R-reduce)

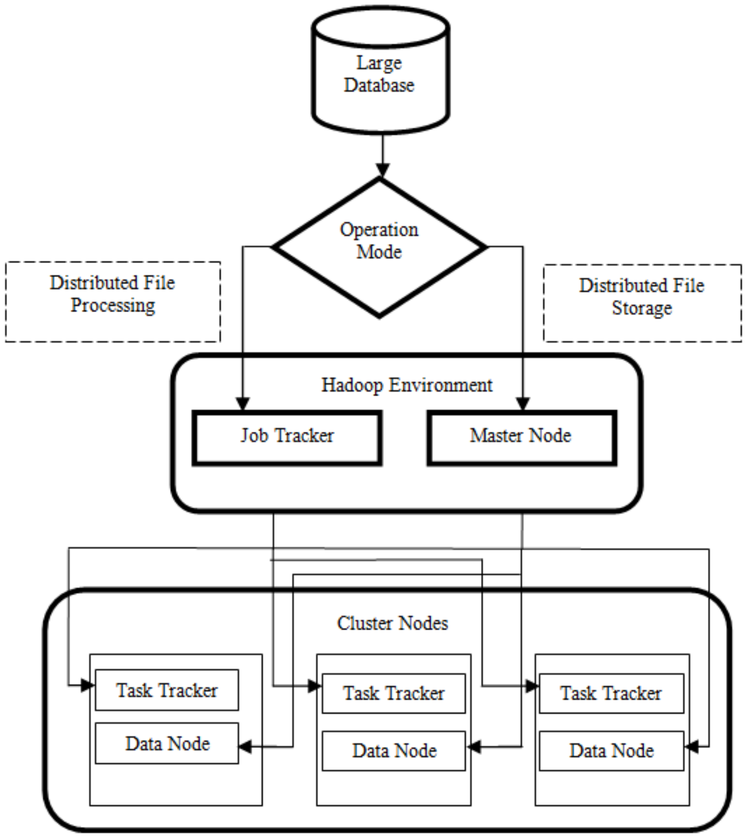


Fig. 2: Hadoop environment

denotes the algorithms, first weighted apriori then hash-t apriori. The existing weighted apriori (Wang *et al*., 2000) is applied in this model which generalizes the traditional model where items are assigned weights. Weighted support is calculated based on the costs to items as well as transactions. A good transaction, which is highly weighted should contain many good items; at the same time, a good item should be contained by many good transactions. This is the main principle of WARM. This study uses this w-apriori algorithm in a Map reduced Hadoop framework using the model depicted as in Fig. 2.

In the new model synthetic data is distributed in a HDFS file system and map reduce processing is being started using weighted apriori and hash-t apriori algorithms (which are all given in the proceeding topics) in the steps like preprocessing, map function and reduce.

Here retail data set has been distributed to various cluster nodes and Map task has been applied and using reduce task the result has been consolidated and taken back. Weighted and hash-t algorithms are applied as map tasks and results are retrieved from the cloud vm-ware environment. This study is mainly concentrating on association rule mining on a hadoop environment and to analyze how it works. Data has been splitted into 4 nodes as primary, secondary and 2 more nodes. The Fig. 2 and 3 shows how the structure of the work is taking place in effect. In the literature survey being conducted these two algorithms outperforms well compared to other algorithms, so they both have been taken for this experiment.

A transaction may contain many items with a non-negative profit. So it is very useful to find out such items. The profit of items and cross-selling effects are very important factors. The method used here eliminates many items which are all not very strongly associated. Hubs (i) is introductory for sales of other items j (i->j). -intended to persuade someone to purchase something for the first time for sales of other items j (Wang *et al*. 2000).

Authorities (j) means necessary for sales of other items i (i->j). This is a must for high frequent item for sales of other items. Hub is calculated by the very first items appearing for transactions. Weight is the process of finding the number of HITS of hub items.

Weighted apriori algorithm:

```
1)  Initialize $auth(i)$ to 1 for each item $i$
2)  for ($l = 0$; $l<num\_it$; $l$++) do begin
3)      $auth'(i) = 0$ for each item $i$
4)      for all transaction $t \in D$ do begin
```

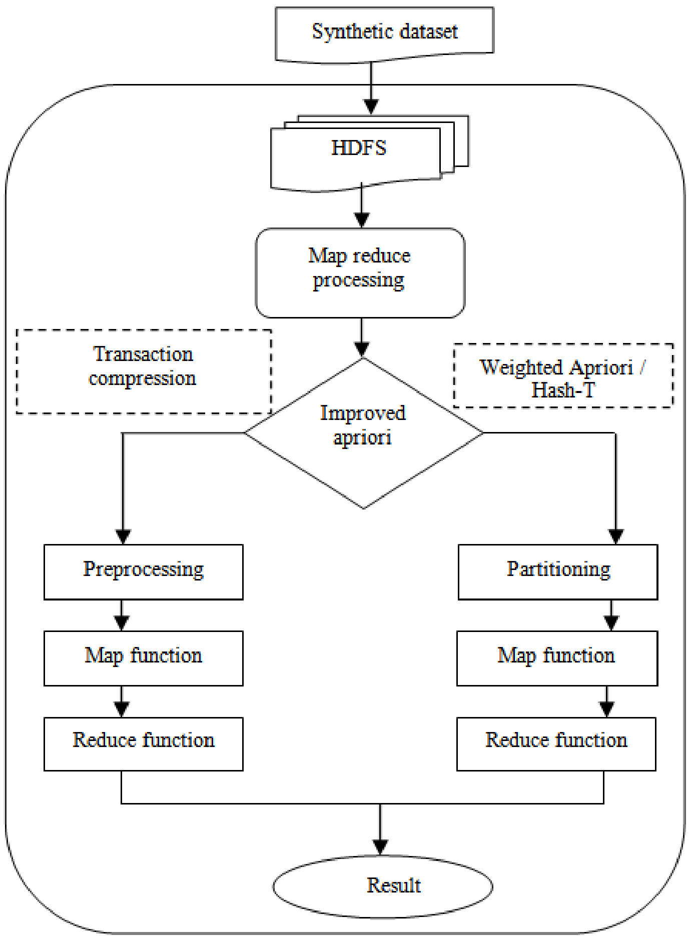


Fig. 3: Weighted apriori and hash-t apriori-map reduce process

```
5)          $hub(t) = \Sigma_{i:i\in t}\, auth(i)$
6)          $auth'(i) += hub(t)$ for each item $i \in t$
7)      end
8)      $auth(i) = auth'(i)$ for each item $i$, normalize $auth$
9)  end
10) $L_1 = \{\{i\}: wsupp(i) \geq minwsupp\}$
11) for ($k = 2$; $L_{k-1} \neq \emptyset$; $k$++) do begin
12)     $C_k$ = apriori-gen ($L_{k-1}$)
13)     for all transactions $t \in D$ do begin
14)         $C_t$ = subset ($C_k$, t)
15)         for all candidates $c \in C_t$ do
16)             c.wsupp += hub (t)
17)         H += hub (t)
18)     end
19)     $L_k = \{c \in C_k | c.wsupp/H \geq minmsupp\}$
20) end
21) Answer = $\cup_k L_k$
```

Algorithm being used for w-apriori (Sun and Bai, 2008) is given in above algorithm.

The algorithm for Hash-T (Grudziński and Wojciechowski, 2009) which reduces candidate-2 itemsets with bucket count is given below.

Algorithm for hash-T:

- Scan all the transactions. Create possible 2-itemsets
- Let the Hash table of size 8
- For each bucket assign an candidate pairs using the ASCII values of the itemsets
- Each bucket in the hash table has a count, which is increased by 1 each item an item set is hashed to that bucket
- If the bucket count is equal or above the minimum support count, the bit vector is set to 1. Otherwise it is set to 0
- The candidate pairs that hash to locations where the bit vector bit is not set are removed
- Modify the transaction database to include only these candidate pairs

In a way similar to map-reduce to apriori, w-apriori and Hash-t can be splitted up using map reduce hadoop framework.

Experiments: The study is carried out in the following method. Hadoop is implemented with 4 nodes in

VMWare virtual machine. Algorithms are coded in eclipse platform. Retail data set is retrieved from http://fimi.ua.ac.be/data/retail.dat. having 1, 76, 324 transactions and 16470 items. The execution is carried out for both the w-apriori and hash-t algorithms and comparison charts are given below. In weighted apriori weight has been considered as a relevance of the item with other items while considering overall unique transactions. Minimum threshold is taken as 0.005. Hash node count is taken as 20.

RESULTS AND DISCUSSION

Graphs show the number of transaction selected in each step and also shows the performance parameters like (Memory, Time, Candidate count etc.), among the two algorithms Hash T gives a better performance in time, candidate count but hash-t occupies memory because of node count. Figure 4 shows the number of selected items in each transaction in w-apriori.

Figure 5 shows metric Parameters, that is metric parameter:

- Means Trans action length ((1, 2, 3, 4, 5) means length = 5)
- Means Memory (Execution memory)
- Means Time (Time to Execute)

During a particular execution hash tree uses memory usage of 140 MB and total time 2400 msec, Weighted Apriori uses 100 MB and total time 2200 msec. The following Table 1 shows the number of association rules generated in each of the iterations for each algorithm.

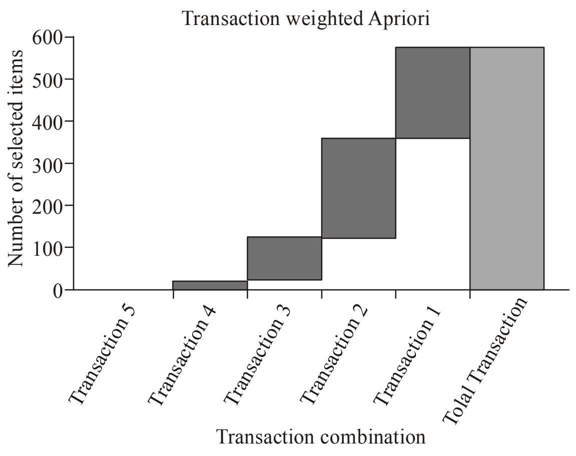


Fig. 4: Number of selected items in each transaction

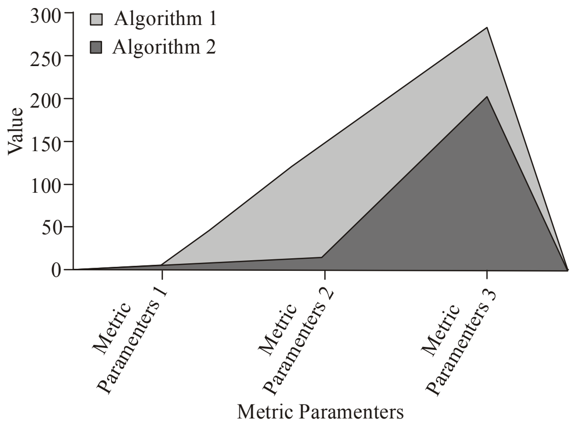
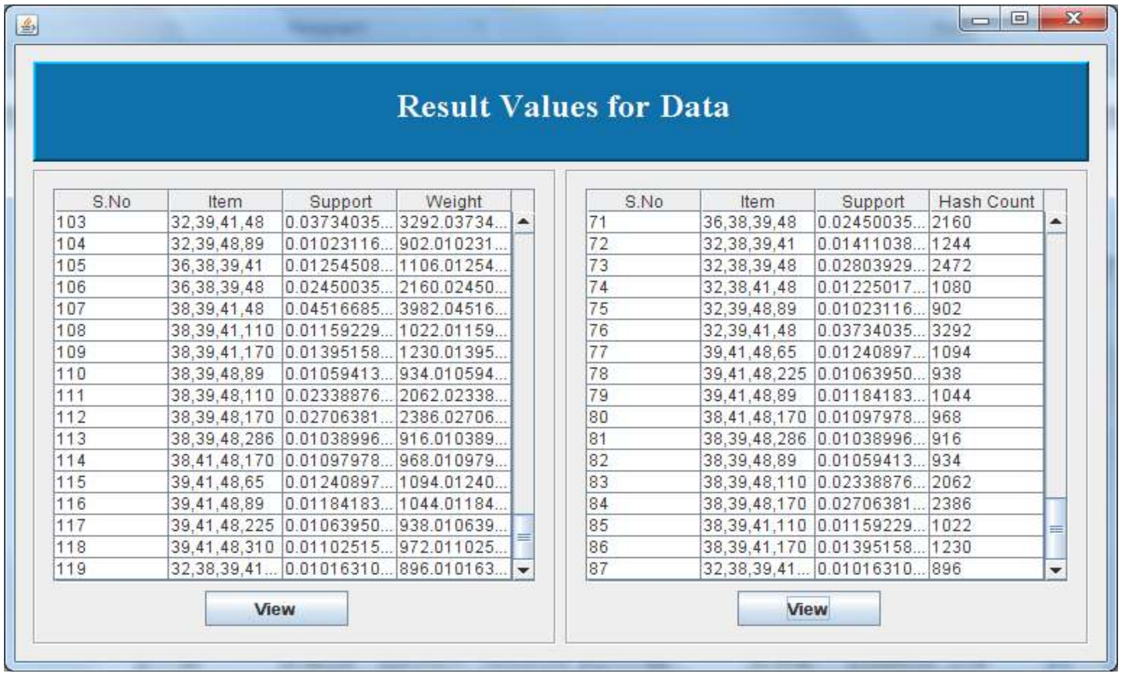


Fig. 5: Metric parameters for each algorithm



S.No	Item	Support	Weight
103	32,39,41,48	0.03734035...	3292.03734...
104	32,39,48,89	0.01023116...	902.010231...
105	36,38,39,41	0.01254508...	1106.01254...
106	36,38,39,48	0.02450035...	2160.02450...
107	38,39,41,48	0.04516685...	3982.04516...
108	38,39,41,110	0.01159229...	1022.01159...
109	38,39,41,170	0.01395158...	1230.01395...
110	38,39,48,89	0.01059413...	934.010594...
111	38,39,48,110	0.02338876...	2062.02338...
112	38,39,48,170	0.02706381...	2386.02706...
113	38,39,48,286	0.01038996...	916.010389...
114	38,41,48,170	0.01097978...	968.010979...
115	39,41,48,65	0.01240897...	1094.01240...
116	39,41,48,89	0.01184183...	1044.01184...
117	39,41,48,225	0.01063950...	938.010639...
118	39,41,48,310	0.01102515...	972.011025...
119	32,38,39,41...	0.01016310...	896.010163...

S.No	Item	Support	Hash Count
71	36,38,39,48	0.02450035...	2160
72	32,38,39,41	0.01411038...	1244
73	32,38,39,48	0.02803929...	2472
74	32,38,41,48	0.01225017...	1080
75	32,39,48,89	0.01023116...	902
76	32,39,41,48	0.03734035...	3292
77	39,41,48,65	0.01240897...	1094
78	39,41,48,225	0.01063950...	938
79	39,41,48,89	0.01184183...	1044
80	38,41,48,170	0.01097978...	968
81	38,39,48,286	0.01038996...	916
82	38,39,48,89	0.01059413...	934
83	38,39,48,110	0.02338876...	2062
84	38,39,48,170	0.02706381...	2386
85	38,39,41,110	0.01159229...	1022
86	38,39,41,170	0.01395158...	1230
87	32,38,39,41...	0.01016310...	896

Fig. 6: A sample screen showing rules and its support, weight and hash count at a particular iteration

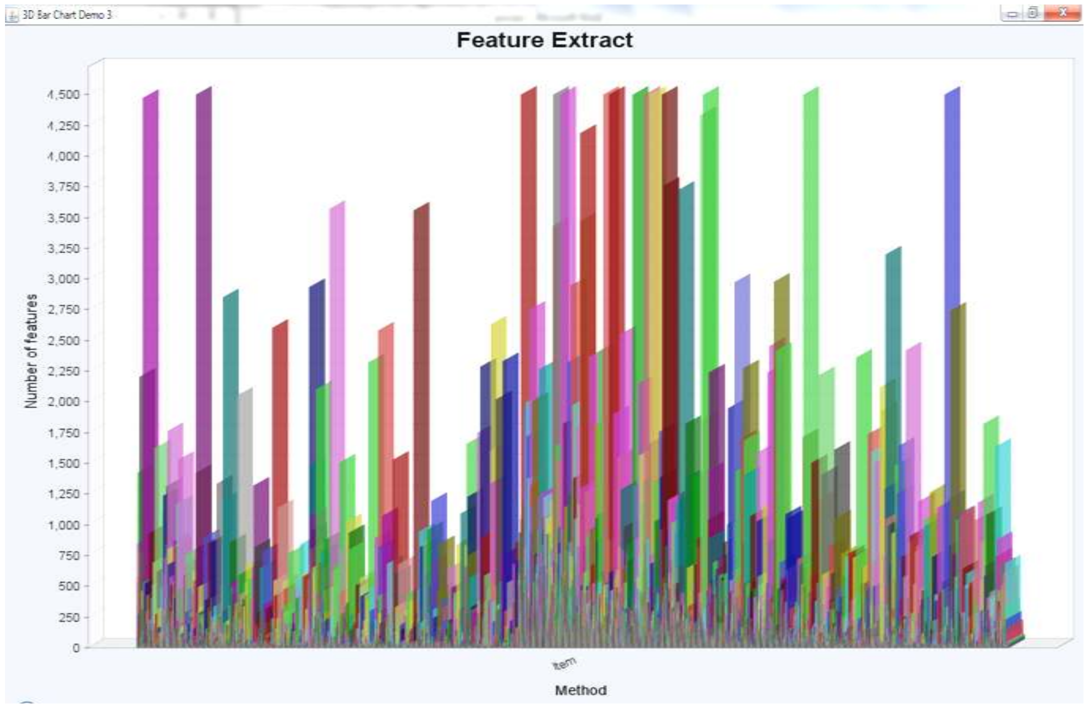


Fig. 7: Feature extraction of all the 16470 items

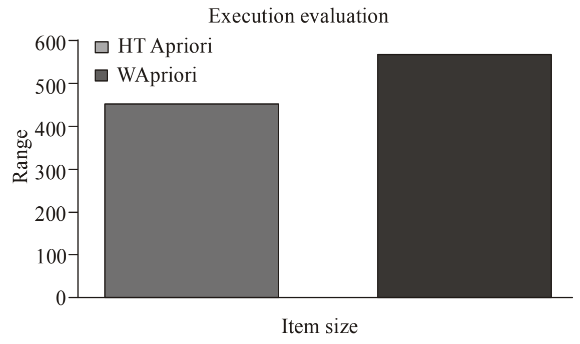


Fig. 8: Item size variation

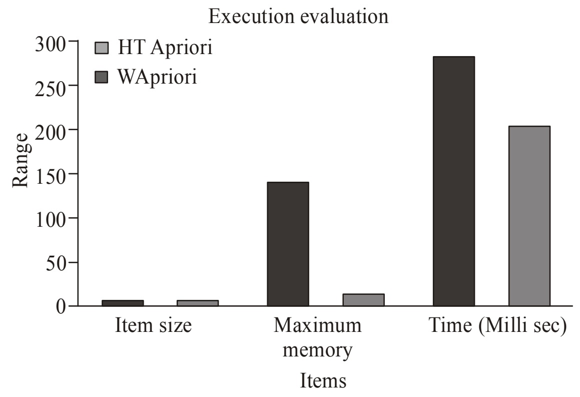


Fig. 9: Execution environment for HT apriori and W-apriori

Though Hash-t gives less number of rules than w-apriori (599 rules) those rules are said to be the best ones than w-apriori. The results of support, weight hash count produced by both the algorithms is given in the following Fig. 6. The occurrence of each item in the total transactions is given in the Fig. 7.

The item size variation of both the algorithms are given in the Fig. 8 in which there is a variation between both the algorithms.

Benefits: Execution time is getting improved (Fig. 9) because in normal apriori the process will happen in the single node, but in hadoop the process will happen parallel in the four nodes using map-reduce schema.

Timing and Memory of weighted apriori is better than hash tree due to the complexity. But number of candidate set generation will be less for hash tree as compared to weighted apriori. Our hash tree will be having the complex execution due to this complexity, time and memory will be high.

CONCLUSION

This study gives a best insight on various techniques of association rule mining algorithms with respect to w-apriori, hash-t, map reduce, hadoop. And also describes the work carried out with a retail data set on hadoop vmware platform using w-apriori and hash t apriori algorithms and compares the performances. The result comparison between w-apriori and hash-t says w-apriori performs well and gives more number of association rules, hash t gives better rules, but occupies more memory and time.

REFERENCES

Agrawal, R. and R. Srikant, 1994. Fast algorithms for mining association rules. Proceedings of the 20th VLDB Conference (VLDB '94). Santiago, Chile, pp: 487-499.

Grudziński, P. and M. Wojciechowski, 2009. Integration of candidate hash trees in concurrent processing of frequent itemset queries using apriori. Control Cybern., 38(1).

Kambatla, K., A. Pathak and H. Pucha, 2009. Towards optimizing hadoop provisioning in the cloud. Proceedings of the 2009 Conference on Hot Topics in Cloud Computing (HotCloud'09), Article No. 22.

Li, J., P. Roy, S.U. Khan, L. Wang and Y. Bai, 2012. Data mining using clouds: An experimental implementation of apriori over mapreduce. Proceeding of the 12th IEEE International Conference on Salable Computing and Communication (ScalCom, 2102). Changzhou, China.

Paul, S. and V. Saravanan, 2008. Hash partitioned apriori in parallel and distributed data mining environment with dynamic data allocation approach. Proceeding of International Conference on Computer Science and Information Technology (ICCSIT '08), pp: 481-485.

Sun, K. and F. Bai, 2008. Mining weighted association rules without reassigned weights. IEEE T. Knowl. Data En., 20(4): 489-495.

Wang, W., J. Yang and P.S. Yu, 2000. Efficient mining of weighted association rules. Proceeding of ACM KDD 2000. Boston, MA, USA, pp: 270-274.

Yang, X.Y., L. Zhen and F. Yan, 2010. MapReduce as a programming model for association rules algorithm on hadoop. Proceeding of 3rd IEEE International Conference on Information Sciences and Interaction Sciences, pp: 99-102.