IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 4, Ver. VI (Jul.-Aug. 2016), PP 01-10
www.iosrjournals.org

Novel Hybrid k-D-Apriori Algorithm for Web Usage Mining

Foram Shah[1], Joanne Gomes[2]
[1](Computer engineering, St. Francis Institute of Technology, India)
[2](Information technology, St. Francis Institute of Technology, India)

Abstract: *The Web usage mining is a branch of web mining in which by clustering the datasets, frequently accessed patterns can be obtained for betterment of social portals, websites. The divisive analysis is one of the types of hierarchical method of data clustering that is used to separate each dataset from the clustered data depending on previous small clusters. Apriori & k-Apriori are commonly used hierarchical algorithms for web usage mining but they are less efficient for mining dynamic item sets such as twitter dataset. Recently D-Apriori algorithm has been proposed in the literature for mining dynamic dataset to find the frequently accessed web pages from web log database. The disadvantage of D-Apriori is that, it takes more execution time compared to k-Apriori. This research work proposes a new hybrid k-D-Apriori algorithm that reduces the execution time, improves frequent pattern generation, works efficiently with dynamic datasets and gives improved association rule generation as compared to D-Apriori algorithm.*

Keywords: *Apriori, Association rule mining, frequent item set, D-Apriori, k-Apriori, k-D-Apriori.*

I. Introduction

Data mining is approach to analyze and summarize the large data in to the required information. Data mining is used in web mining to conceptualize patterns from web. How web miming is different than data mining can be discussed with the help of parameters such as scale, access, structure of data [1]. Web mining is one of the broader areas for research work. It includes mining of web content, web structure and web usage [1], [2]. In web content mining, mining of web page content is done. It includes data such as text, image, audio, video, structured records. Web structure mining is recognizing interrelation of information from web which includes hyperlinks, document structures etc. In web usage mining usage patterns from web server log, application server log and web usage data gets discovered in order to gain better performance of the web site. To perform frequent pattern mining on web log data different approaches such as Apriori, frequent pattern growth, and vertical data format approach are used [3], [4]. In order to improve the pattern mining, clustering is used. Clustering is one of the steps of web mining process to group a set of objects in such a way that objects in the same group are similar to each other than to those in other groups. It is one of the unsupervised learning method where no training data is given. Clustering algorithms are of various types such as Partitioning, Density Base, Hierarchical, Grid base, Model based, Evolutionary & fuzzy [3]. Hierarchical clustering is a method of cluster analysis which seeks to build a hierarchy of clusters. Hierarchical clustering is further divided into Agglomerative (a "bottom up") approach and Divisive (a "top down") approach [4]-[6].

For hierarchical clustering Apriori algorithm performs mining of frequent item sets by generating candidate items where frequent item sets are mined with the help of some support value. For this, database gets scanned from top to bottom [6], [7]. Disadvantage of Apriori algorithm is, it requires many database scans and it is less efficient with dynamic database [5]. To overcome multiple database scans k-Apriori algorithm was introduced in [8] where Apriori was used to find frequent item sets but the given input was passed through wiener-transformation. This was followed by a k-means algorithm that helped to reduce multiple database scans and to improve the performance of Apriori. But the problem of dynamic data acceptance still existed and hence to solve it, D-Apriori algorithm was introduced in [9]. D-Apriori algorithm treats data in terms of attributes and queries, converts them in the form of binary matrix and then uses Apriori to form frequent set mining. Web mining of dynamic data sets can be further improved in terms of generating less frequent item sets, less association rules and less execution time [9]-[11].

This paper proposes a new hybrid k-D-Apriori approach that is a combination of Apriori, k-Apriori and D-Apriori algorithms. Implementation of the said algorithm is done by using eclipse enterprise edition IDE. It is observed that the proposed k-D-Apriori algorithm gives better performance related to mining dynamic frequent item sets and is advantages over previous algorithms in terms of less execution time and less number of iterations. The paper is organized as follows.

Section II presents related work of mining approach. Section III discusses new hybrid k-D-Apriori approach. Section IV is related with simulation environment and results. Section V assimilates the simulation results. Section VI gives observation and comparison of work done. Section VII concludes the paper.

II. Related work

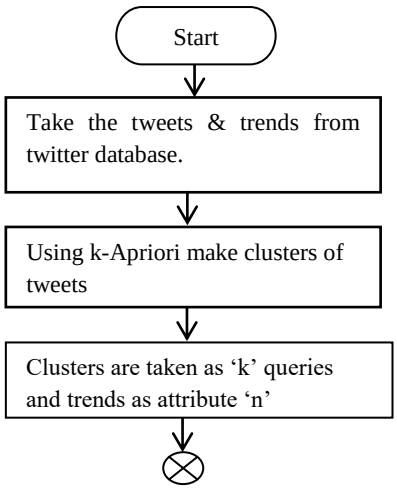
As discussed earlier Apriori algorithm is easy to implement for association rule generation, but it assumes the transaction database as memory occupier as given in [5]. It requires many database scans and does not work efficiently with dynamic data. Also the computational efficiency of the algorithm is very less. To overcome the problem related to multiple database scans of Apriori, new algorithm was proposed, named k-Apriori algorithm [5], [8]. However, even k-Apriori is less efficient with dynamic data set as per [5], [8]. Hence to work with dynamism, a new algorithm named D-Apriori (Dynamic Apriori) was proposed in [9].

In D-Apriori algorithm transactions were added to the existing database at run time. It overcomes the limitation of Apriori and k-Apriori algorithm and supports dynamic datasets [9]. Yet, there is further scope of improvement in performance of D-Apriori algorithm in terms of execution time and frequent set generation. In summery Apriori, k-Apriori, and D-Apriori are existing algorithms, each one of them have shown improvements in the performance over its prior versions.

To overcome problems associated with D-Apriori a new algorithm k-D-Apriori is proposed in this research work. Twitter datasets are used to perform frequent item set mining and generate association rules. Reason for using twitter data is that, tweets happen at the "velocity of thinking" which makes it very dynamic and attractive. Obtaining twitter data from anywhere in the world is comparatively easy for utilization in real time. Twitter data is well described and its API's are easy to access. Also it is available in an acceptable format for data analysis. Compared to other APIs, terms of use for twitter's data are liberal and tweets are open to all and accessible to anyone.

III. New Hybrid k-D-Apriori Approach

The proposed new k-D-Apriori algorithm has potential to improve performance related to dynamic frequent item set mining and to generate association rules. It is implemented for mining dynamic web data such as Facebook and twitter in an efficient way. In this research work, twitter data sets of 1000, 5000, 10000 records with different trends are used. Here trends are current activities going around the globe and the tweets are comments of people under discussion. The required data is collected using twitter 4J API. Data is stored using MySQL, under the table 'newtrend'. It contains columns such as Trend, ScreenName, Created_At, Tweetid and Tweet which includes latest records synchronized from twitter database using 4J API. The flow of k-D-Apriori algorithm is given in Fig. 1.



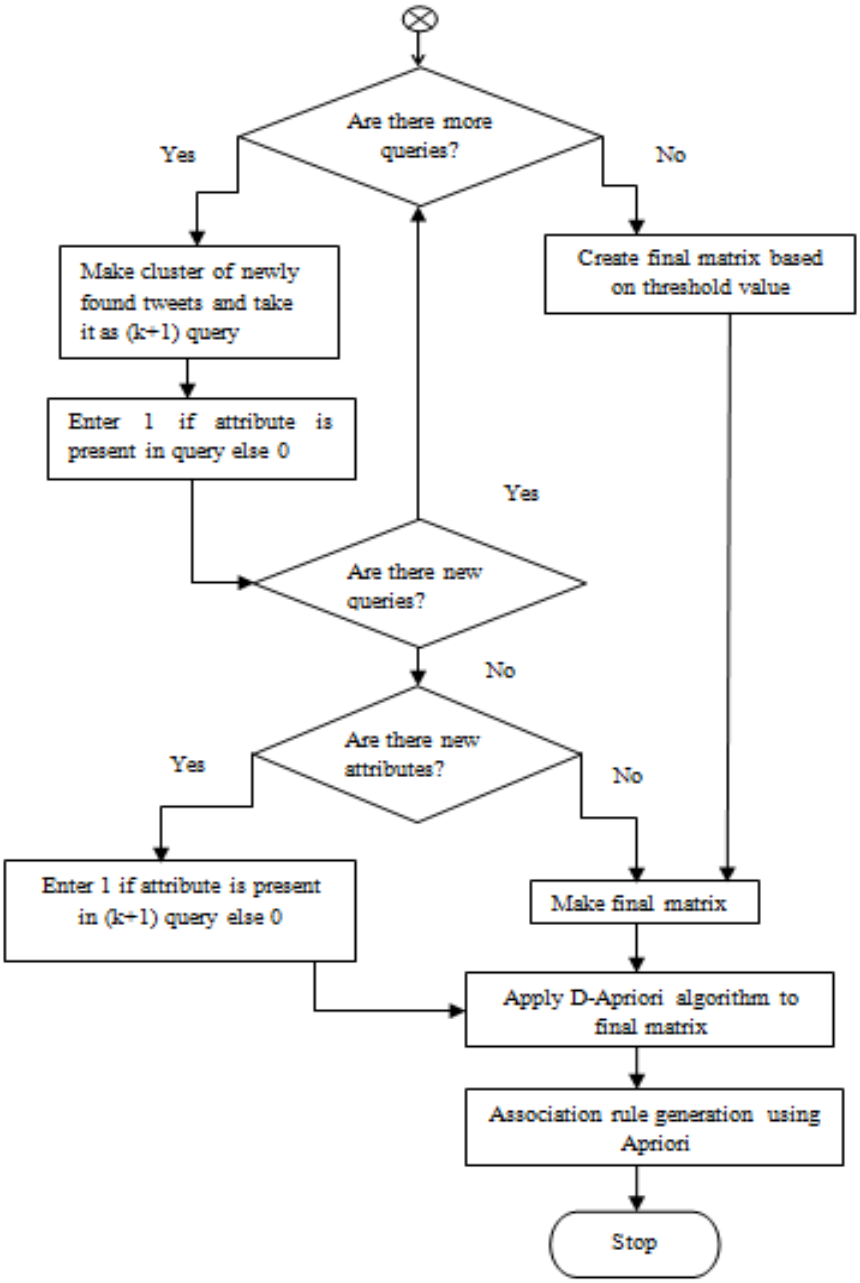


Fig. 1: Flow of k-D-Apriori algorithm

Algorithm shows procedure of clustering, mining data items, final frequent item set generation and overall working of k-D-Apriori algorithm for static as well as dynamic data set. Initially 'n' number of attributes are taken which are number of trends considered. In next step, values of all 'n' attributes are collected. Similarly number of tweets and their values are collected from database in terms of attributes and queries. Clustering process of k-Apriori algorithm [5] is applied and its output is used to form final matrix (binary) of frequent items using D-Apriori algorithm [9]. Final matrix contains value as '1' if 'n' attribute is present in query 'k'. Else, it will be '0'. Further, k-D-Apriori will check for dynamically added tweets and trends and update the content of final matrix accordingly. Binary matrix will make number of database scan lesser as well as shorter time span is required for dynamic data without the need for entire database scans as required by existing algorithms. Apriori algorithm [5] is then applied to the data set given by final matrix to generate association rules in order to show interdependency of trends with each other.

Final output of k-D-Apriori algorithm is nothing but item sets which are frequently accessed these sets are sets of trends on which highest tweets are present. On the basis of support and confidence value, association rules get generated which give interrelationship of trends. It gives clear idea about the related trends which are combined for user to give their comments.

IV. Simulation Environment

The said algorithm was implemented using IDE eclipse Luna edition on the personal computer with Intel ® Pentium, ® CPU N3540, @ 2.16GHz, 4GB RAM with 64 bit operating system and x64- based processor. Wampserver was used for database connectivity on windows platform. The simulation results were carried out using simulation parameters shown in Table 1.

Table 1: Simulation parameters

Parameters	Values
Number of records	1000, 5000, 10000, online set of dynamic data
Support	2, 3, 4, 5
Confidence	0.3, 0.5

The data accessed from twitter was stored in table 'newtrend' of MySQL database 'twittertrend'. After applying k-D-Apriori algorithm, the most frequently used trends were obtained. These trends were further used to generate association rules that showed interdependency of the trends. The following Table 2 shows the performance parameters of k-D-Apriori algorithm using twitter data set of 1000 records, support=3 and confidence=0.3.

Table 2: k-D-Apriori output parameters

Parameters	Values
Execution Time(in sec)	0.017
Frequent Sets obtained	27
Association Rules generated	34
Number of Iterations	3

The aim of k-D-Apriori algorithm is to give improved performance with less execution time and more accurate association mining. (1) And (2) indicate rules used in k-D-Apriori algorithm for calculation of support and confidence values which further obtain association rules [11]. For example consider items A and B. To calculate support of A → B (1) is used [11]. To calculate confidence of A → B equation (2) is used [11]. Item sets with greater value than minimum support and confidence are considered and taken as final association rules. Association rules thus generated show the dependency of trends among each other. Equations (2) and (3) indicate formula used to calculate precision and recall values respectively as performance parameters of k-D-Apriori algorithm.

$$\text{Support}(A \rightarrow B) = \frac{\text{No. of Transactions containing both A \& } B}{\text{Total No. of Transactions}} \quad (1)$$

$$\text{Confidence}(A \rightarrow B) = \frac{\text{No. of Transactions containing both A \& } B}{\text{Transactions containing only A}} \quad (2)$$

$$\text{Precision} = \frac{\text{total rules generated by algorithm}}{\text{possible rules can be generated}} \quad (3)$$

$$\text{Recall} = \frac{\text{total rules generated by algorithm}}{\text{total relevant or actual rules can be generated}} \quad (4)$$

V. Simulation Results

This section assimilates the simulation results obtained using Apriori algorithm, k-Apriori algorithm, D-Apriori algorithm and k-D-Apriori algorithm.

Case 1: Performance of Apriori algorithm

Tables 3 and 4 show performance of Apriori algorithm for different support values of 2, 3, 4, 5 and confidence values with 0.3 and 0.5. The performance of algorithm is measured in terms of Execution time, frequent set obtained, Association rules generated, Number of Iterations. Figs. 2 and 3 indicate these performance parameters.

Table 3: Apriori performance using confidence= 0.3

Parameters	**Value**			
Support	2	3	4	5
Execution time (in sec)	2.655	0.078	0.068	0.086
Frequent sets obtained	97	83	82	64
Association rules generated	107	85	85	44
Number of Iterations	5	5	5	5

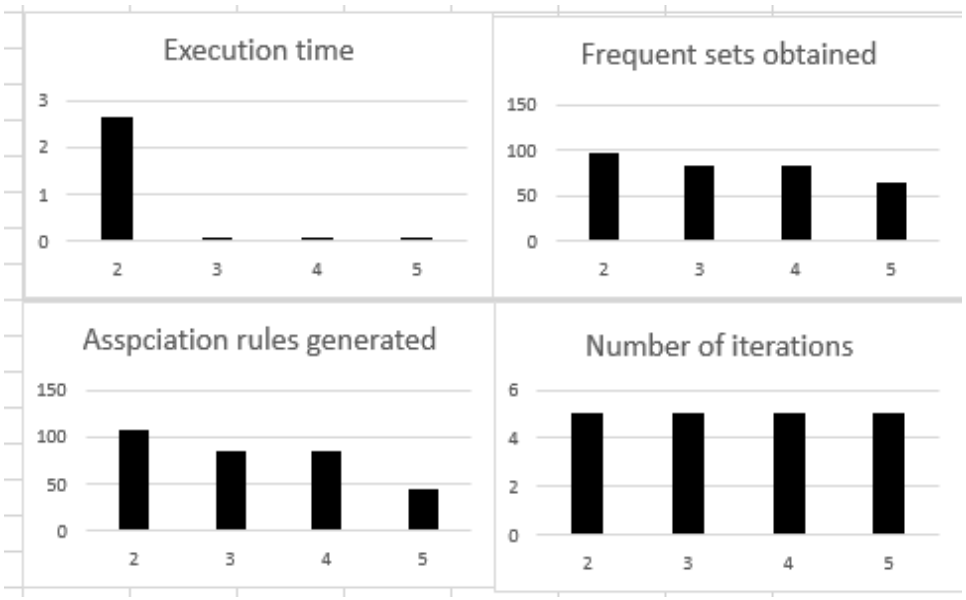


Fig. 2 Apriori performance using confidence= 0.3

Table 4: Apriori performance using confidence= 0.5

Parameters	Value			
Support	2	3	4	5
Execution time (in sec)	0.09	0.992	0.063	0.089
Frequent sets obtained	97	83	82	64
Association rules generated	88	67	67	26
Number of Iterations	5	5	5	5

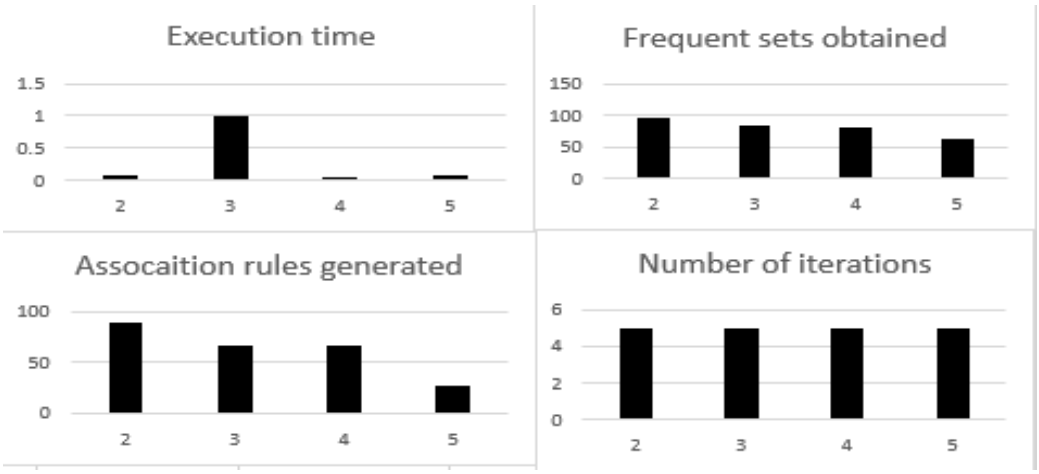


Fig. 3 Apriori performance using confidence= 0.5

Case 2: Performance of k-Apriori algorithm

Tables 5 and 6 show performance of k-Apriori algorithm for different support values, where frequent item sets are in decreasing order. Association rules generated for higher support value are less. Number of iterations for support value 2, 3, 4 are constant and in decreasing order for higher support value 5.

Table 5: k-Apriori performance using confidence= 0.3

Parameters	Value			
Support	2	3	4	5
Execution time (in sec)	0.205	0.063	0.059	0.114
Frequent sets obtained	67	63	62	50
Association rules generated	33	31	31	6
Number of Iterations	5	5	5	3

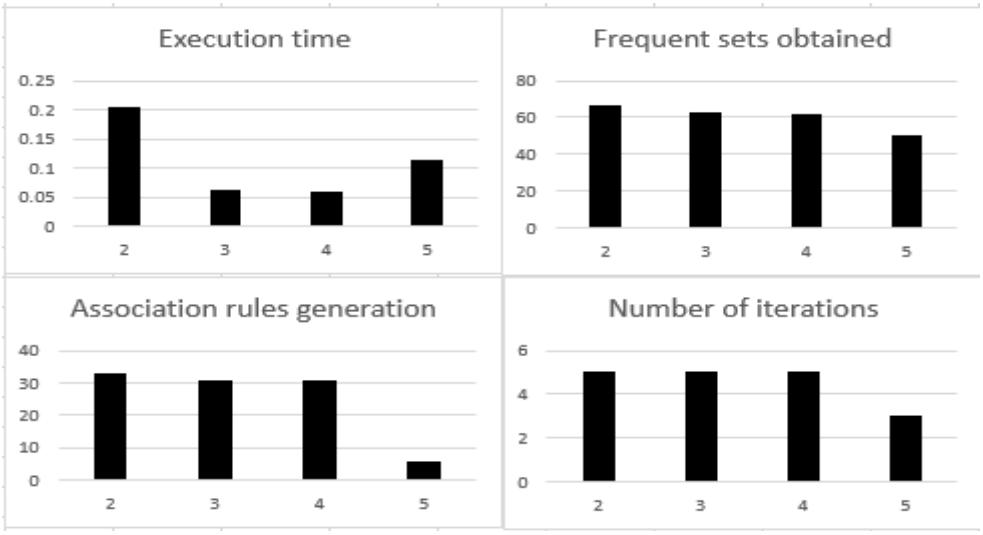


Fig. 4 k-Apriori performance using confidence= 0.3

Table 6: k-Apriori performance using confidence= 0.5

Parameters	Value			
Support	2	3	4	5
Execution time (in sec)	0.065	0.181	0.056	0.058
Frequent Sets obtained	67	63	62	50
Association rules generated	30	28	28	3
Number of Iterations	5	5	5	3

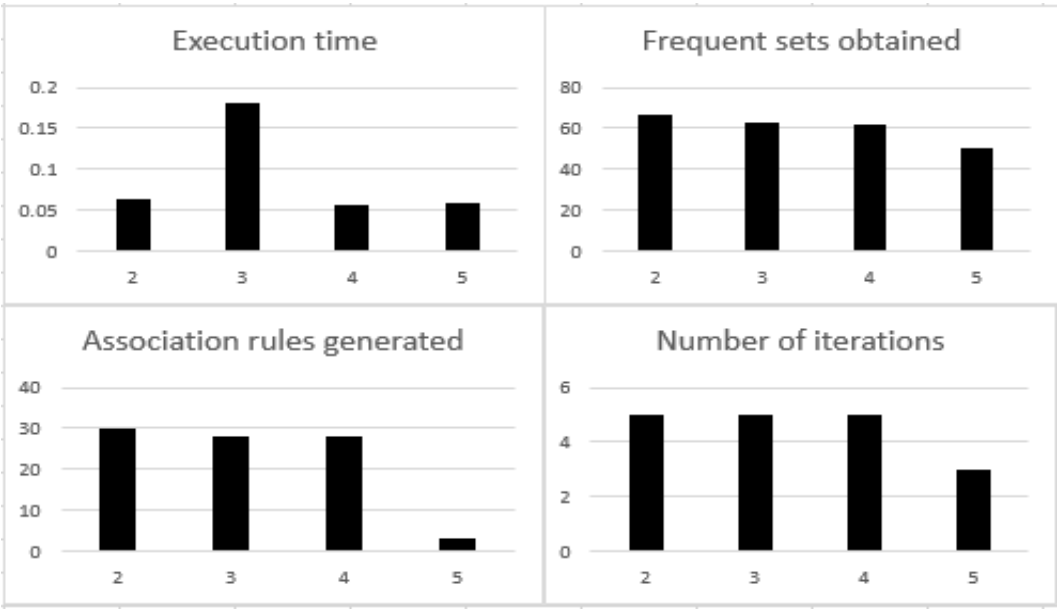


Fig. 5 k-Apriori performance using confidence= 0.5

Case 3: Performance of D-Apriori algorithm

Tables 7 and 8 show performance of D-Apriori algorithm for different support and confidence values. Here frequent item sets are decreasing as support value is increasing. Number of association rules generated and iterations taken for generating frequent set are constant for all support value.

Table 7: D-Apriori performance using confidence= 0.3

Parameters	Value			
Support	2	3	4	5
Execution time (in sec)	0.919	0.478	0.451	0.423
Frequent sets obtained	50	36	36	18
Association rules generated	4	4	4	4
Number of Iterations	3	3	3	3

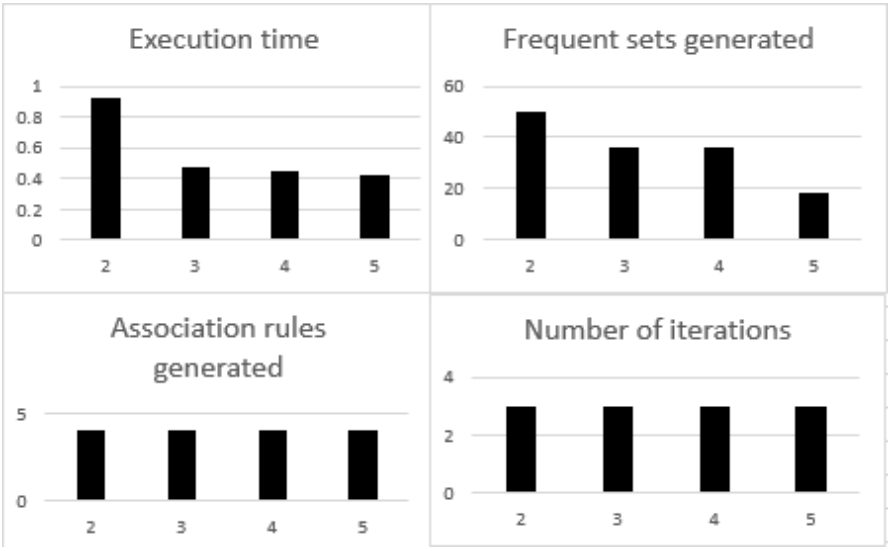


Fig. 6 D-Apriori performance using confidence= 0.3

Table 8: D-Apriori performance using confidence= 0.5

Parameters	Value			
Support	2	3	4	5
Execution time (in sec)	0.345	1.507	0.319	0.315
Frequent sets obtained	50	36	36	18
Association rules generated	4	4	4	4
Number of Iterations	3	3	3	3

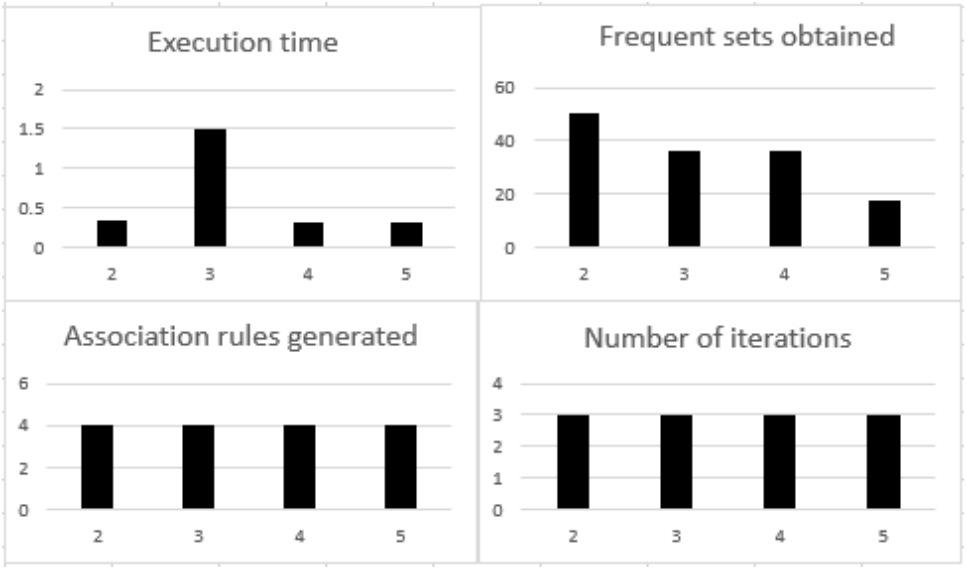


Fig. 7 D-Apriori performance using confidence= 0.5

Case 4: Performance of k-D-Apriori algorithm

Tables 9 and 10 show performance of k-D-Apriori algorithm for different support values where frequent sets and association rules generated, Number of iterations are decreasing with increasing support values. Execution time is lower for less support values whereas it further changes in inconclusive order.

Table 9: k-D-Apriori performance using confidence= 0.3

Parameters	Value			
Support	2	3	4	5
Execution time (in sec)	0.044	0.017	0.021	0.029
Frequent sets obtained	45	27	27	4
Association rules generated	59	34	34	0
Number of Iterations	4	3	3	2

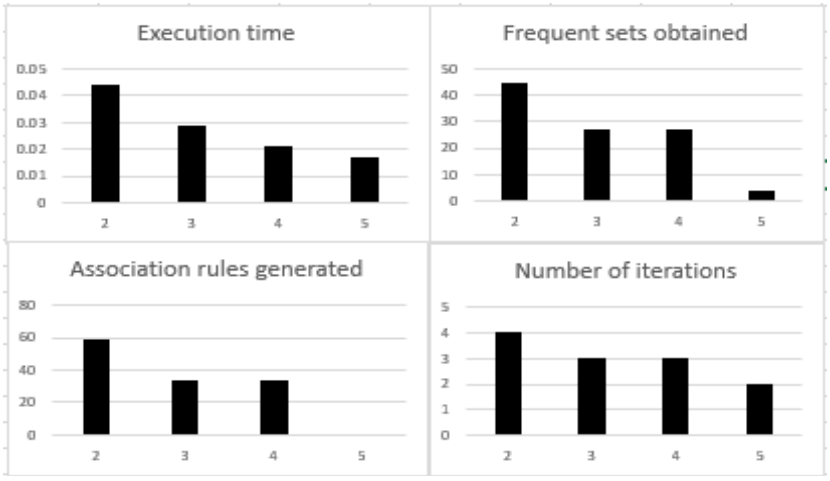


Fig. 8 k-D-Apriori performance using confidence= 0.3

Table 10: k-D-Apriori performance using confidence= 0.5

Parameters	Value			
Support	2	3	4	5
Execution time (in sec)	0.018	0.038	0.014	0.009
Frequent Sets obtained	45	27	27	4
Association Rules generated	57	23	23	0
Number of Iterations	4	3	3	2

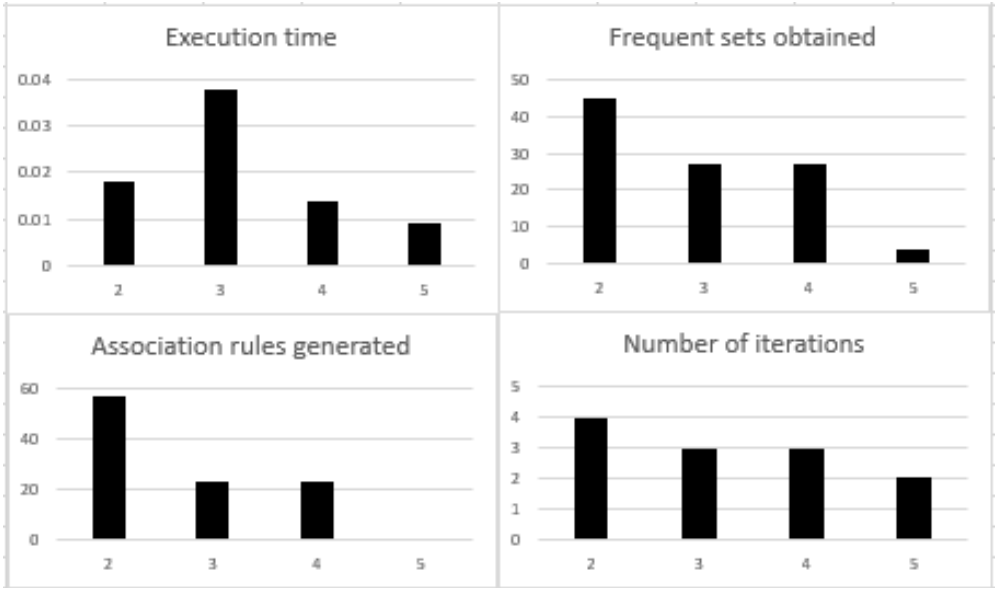


Fig. 9 k-D-Apriori performance using confidence= 0.5

The frequent sets and association rules generated shows descending trend against the increasing support values. Moreover, number of iterations required for generating frequent sets are reduced.

VI. Observations

In the tables 3 to 10 it has been observed that for Apriori algorithm all the parameters are in decreasing form. For k-Apriori performance is improved in form of execution time and frequent set generation. D-Apriori works efficiently for dynamic dataset and its execution time is further reduced than k-Apriori. Proposed k-D-Apriori is more improved in terms of execution time, association rules and frequent set generated.

Comparative analysis of proposed k-D-Apriori algorithm:

Table 11 shows comparative analysis of k-D-Apriori. It clearly shows that Apriori algorithm generates more frequent set and association rules compare to others. Since Apriori and k-Apriori is less efficient with dynamic data sets. Main comparison is between D-Apriori and k-D-Apriori algorithm because they both support dynamism. Performance of k-D-Apriori is improved compare to D-Apriori in terms of association rules generation and execution time and frequent pattern generation taken as shown in table 11.

Table 11: Comparisons of Apriori, D-Apriori, k-Apriori and k-D-Apriori

Algorithm	Execution Time(in sec)	Frequent Sets obtained	Association Rules generated	Number of Iterations
Apriori	0.078	83	85	5
D-Apriori	0.478	36	4	3
k-Apriori	0.205	67	33	5
k-D-Apriori	0.017	27	34	3

Fig. 10: Comparisons of Apriori, D-Apriori, k-Apriori and k-D-Apriori Precision and recall analysis:

Table 11 compares the precision and recall rate of above mentioned algorithms simulated in this paper. The result shown in table 11 are evaluated using precision and recall formulas given in Eqns. (3) and (4). Graph in Fig. 12 show how actual rules generated by k-D-Apriori algorithm are accurate.

Table 11: Precision and Recall calculation

Algorithm	Precision	Recall
Apriori	100	40
D-Apriori	4.70	850
k-Apriori	36.47	109.67
k-D-Apriori	40	100

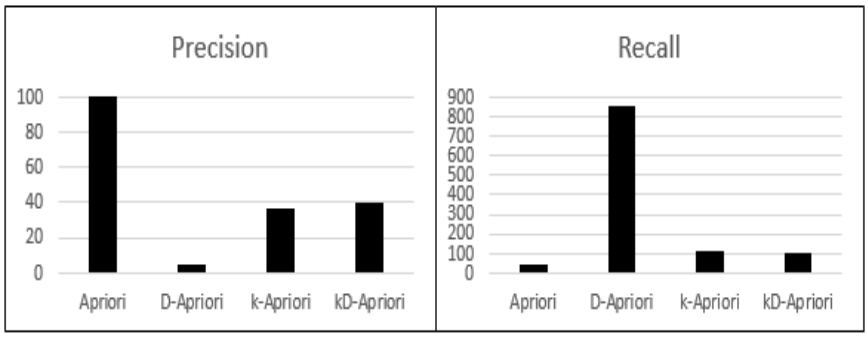


Fig. 12: Precision and recall calculation

Scalability analysis:

In order to check scalability of proposed k-D-Apriori algorithm 3 sets of 1000, 5000, 10000 records are used. Value associated with all parameters are shown in Table 12. Scalability graph of proposed k-D-Apriori algorithm is shown in Fig. 13. It is seen from the graph, that as the number of records increase all the parameter values increase accordingly. Hence we can say that the proposed algorithm is scalable.

Table 12: Scalability comparison

No. of records	Execution Time (in sec)	Frequent Sets obtained	Association Rules generated	Number of Iterations
1000	0.22	27	34	3
5000	0.37	162	441	6
10000	0.445	406	1179	7

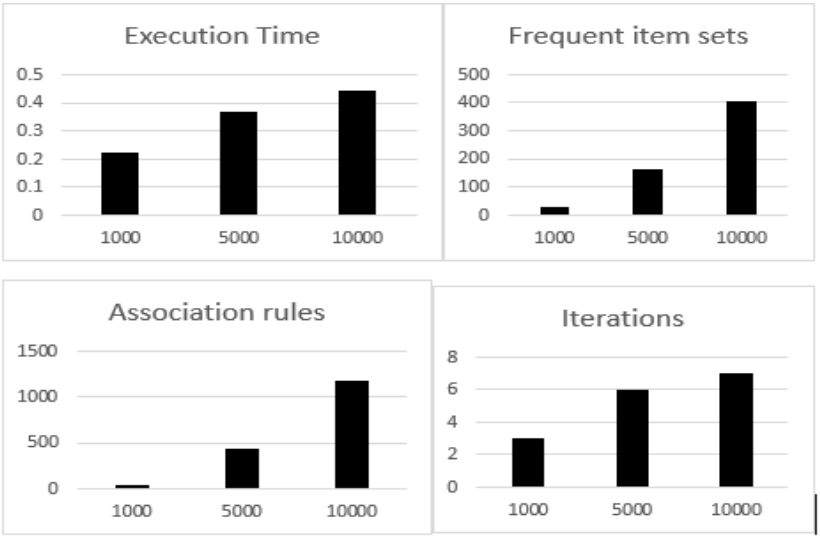


Fig. 13: Scalability Comparison

VII. Conclusion

D-Apriori is recently proposed algorithm for data mining of dynamic item sets. This research work proposed novel k-D-Apriori algorithm for betterment of performance for mining dynamic datasets such as twitter data logs. Comparative analysis of existing algorithms along with proposed hybrid k-D-Apriori algorithm, proves that the proposed algorithm is an effective algorithm for generating efficient frequent item set and association rules from dynamic data set.

Generated association rules, can be used to improve the performance of twitter website by providing trends in such combination that it will lead to increase in tweets, people post in the web site, which will help to attracts more people to use & register with the twitter website.

References

[1] Kavita Sharma, Gulshan Shrivastava, and Vikas Kumar, "Web Mining: Today and Tomorrow", Issues and Challenges in *Proc. Electronics Computer Technology*, Apr. 2011.

[2] DataMining:www.anderson.ucla.edu/faculty/jason.frand/teacher/.../datamining.htm.

[3] Lokesh s., Deepti s., sheetal s., and Khushboo s., "Clustering Techniques: A Brief Survey of Different Clustering Algorithms" *International Journal of Latest Trends in Engineering and Technology, Vol 1, issue 3, pp 82-87 Sep. 2012.*

[4] Gupta, A. Arora, R. Sikarwar, and R. Saxena, N, "Web usage mining using improved Frequent Pattern Tree algorithms", Issues and Challenges in *Proc. Intelligent Computing Techniques (ICIC), Feb. 2014.*

[5] Kumar Ashok, Charlet Annie and M. C. Loraine, "Web Log Mining using K-Apriori Algorithm", *International Journal of Computer Applications, Vol. 41, issue 11, pp. 16-20, Nov. 2012.*

[6] Binhua Liao, "An Improved Algorithm for Apriori", *International Journal of Communications in Computer and Information Science, Vol 51, issue 8 pp. 427-432, Nov. 2009.*

[7] Pranay Bhandari , K.Rajeswari, Swati Tonge, Mahadev Shindalkar,"Improved AprioriAlgorithms – A Survey*", International Journal of Advanced Computational Engineering and Networking, Vol-1, issue-2, pp 2320-2106, Apr. 2013.*

[8] Kumar Ashok, Charlet Annie & M. C. Loraine, "Web Log Mining using K-Apriori Algorithm", *International Journal of Computer Applications, Vol. 41, issue 11, pp 16-20, Mar. 2012.*

[9] Bagga and S.Badal, "D-Apriori: An Algorithm to Incorporate Dynamism in Apriori Algorithm", *International Journal of Computer Applications, Vol. 89, issue 10, pp. 24-28,Mar. 2014.*

[10] R.Kousalyaand and V.Saravanan, "Web Usage Mining Using D-Apriori and DFP Algorithm", *International Journal of Scientific and Engineering Research, Vol. 4, issue 3, pp 12-15, Mar. 2013.*

[11] Suriya and Shantharajah, "A hybrid k-DCI and Apriori algorithm for mining frequent item sets", in *Proc. Circuits, Power and Computing Technologies (ICCPCT)*, Nov.2013.

[12] Agrawal, Rakesh, and Ramakrishnan Srikant. "Fast algorithms for mining association rules." in *Proc. 20th int. conf. very large data bases, VLDB. Vol. 1215, pp 487-499, Sep. 1994.*