International Innovative Research Journal of Engineering and Technology
ISSN NO: 2456-1983

International Conference on Emerging Innovation in Engineering and Technology

ICEIET-2017

A Survey on Frequent Itemset Mining

Nithya.M[1], Kanmani.S[2]

[1]M.Tech, Student,[2]Professor, Information Technology
[1,2]Pondicherry Engineering College,
Puducherry, India.

Abstract:

Frequent itemset mining is the technique used mostly in field of data mining like finance, health care system. We are focusing on methodologies for extracting the useful knowledge from given data by using frequent itemset mining. Most important use of FIM is customer segmentation in marketing, shopping cart analyzes management relationship, web usage mining, and player tracking and so on. The time required for generating frequent itemsets plays an important role. Some algorithms like Apriori, Eclat, FP-Growth are designed, considering only the time factor. Our study includes depth analysis of algorithms and discusses some problems of generating frequent itemsets from the algorithm. We have explored the unifying feature among the internal working of various mining algorithms. The comparative study of algorithms includes aspects like different support values, size of transactions and different datasets.

Keywords: Apriori, Eclat, FIM, FP-Growth.

I. INTRODUCTION

Data mining is the process of extracting useful, potential, novel, understandable, and concealed information from databases that are huge, noisy, and ambiguous. Data mining plays a vital role in various applications in the modern world, such as market analysis, credit assessment, fraud detection, medical, fault diagnosis in production systems, insurance and healthcare, banking and finance, hazard forecasting, Customer Relationship Management (CRM), and exploration of science. Many view data mining as synonymous to Knowledge Discovery from Data (KDD), while others consider data mining as an essential stage in the KDD process. The outline of the KDD process is shown in Figure 1.

The first step is to define a problem from a particular domain that contains appropriate previous knowledge and particular application goals. The second process is choosing an appropriate dataset, which consists of a dataset or concentrates on a subset of variables or data samples on which discovery is to be accomplished. Pre-processing is the third step of the KDD process, which consists of data collecting, data cleaning, and data selection. It is the key step in the KDD process, and it removes noise, outliers, and redundant or irrelevant information, handles missing data fields, and determines DBMS issues, such as types of data, schema, and the

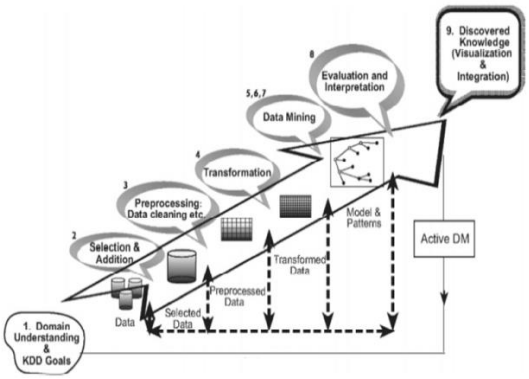

Figure 1. Knowledge Discovery of KDD Process

managing of missing and unknown values. The fourth step is data transformation, which may refer to data reduction and projection; this process helps to discover the most valuable features of the data that is depending on the task and applies dimensionality reduction, such as reducing the number of attributes, attribute values, and tuples, or transformation methods, such as normalization, aggregation, generalization, and attribute construction, to reduce the effective number of variables under

consideration or to find invariant representations for the data.

The fifth step is one of the most important processes for selecting the appropriate function of data mining; it constructs a suitable model derived by the particular data mining algorithm (e.g. association rule mining, classification, summarization, clustering, and regression).

The sixth step is choosing the proper data mining algorithm(s), which includes selecting technique(s) to be used to find the patterns of the data, such as deciding which models may be proper and matching a particular data mining technique with the KDD process. The seventh key step is data mining, which includes discovery of the interesting patterns in the particular assigned dataset, including classification rules, decision trees, regression, clustering, sequence modeling, dependency, and line analysis.

The eighth step is interpretation, which consists of data mining techniques and finding out whether a good clustering or classifying approach must interpret the result of such an approach. If a result cannot be explained properly, it is useless for further application. The last step is utilizing the discovered knowledge, i.e. using a newly discovered set of knowledge for future analysis and the prediction of new models[5].

II. LITERATURE REVIEW

Data mining literature already has sequential and parallel algorithms for finding frequent itemsets although there is need of parallel algorithms which can work on widely used distributed platform, MapReduce because there exit various issues of scalability and performance for existing parallel algorithms in the era of Big Data.

Moens et al.[8] proposed two methods for finding frequent itemset mining for Big Data on MapReduce, First method Dist-Eclat is distributed version of pure Eclat method which optimizes speed by distributing the search space evenly among mappers, second method BigFIM uses both apriori based method and Eclat with projected database that fit in memory for extracting frequent itemsets.

Modification of Apriori on MapReduce has been proposedbyLinetal. Single Pass Counting(SPC),Fixed Pass Counting(FPC) and Dynamic Passes Counting (DPC) which do counting step parallel by distributing dataset to mapper.

Li et al[7] proposed parallel version of apriori based algorithm on MapReduce. Apriori based algorithm does not work on large datasets having long frequent itemsets.

Hammoud has proposed MRApriori approach for finding frequent itemsets by switching between vertical and horizontal layout iteratively which eliminates need of iterative scanning of data. It repeats scanning for other intermediate data which reduces iteration.

III. RELATED WORK

Apriori Algorithm:

Apriori algorithm (Agrawal et al. 1993), is the utmost central and significant algorithm for mining frequent things. All the frequent things in a given database are searched with the help of Apriori algorithm. The keynote of Apriori algorithm is to create several passes on the database. It pays an repetitive procedure called as a breadth-first search which is also known as level wise search over the search area, where k-things are recycled to discover (k+1)things.

WORKING OF APRIORI

- Discovery of all frequent itemsets.

Catch frequent things: Things whose existence in database is more than or equivalent to the smallest help threshold.

i. Generate candidates from frequent things.
ii. Prune the outcomes to identify the frequent itemsets.

- Develop robust association rules from frequent itemsets.

It consist rule which fulfills the minimum support and minimum confidence threshold.

ECLAT Algorithm

Eclat algorithm works on the basis of the Vertical data format. First, the item sets are checked in lexicographic order i.e nothing but depth-first traversal of the prefix tree. The search plan is the same as the general pattern for with canonical forms having the prefix assets and holding a picture-perfect extension rule (just produce canonical extensions). Eclat produces many candidate item sets than Apriori, because it does not accumulate the support of every visited item sets. Like Apriori algorithm pruning technique is cannot used in this algorithm to reduce the candidate itemset. Eclat practices a only vertical transaction demonstration. For this algorithm

no subset checks and no subset generation are desired to calculate the support. The support of item sets is fairly determined by intersecting transaction lists.

FP-Growth Algorithm

The most widespread algorithm for frequent itemset mining is nothing but the FP-Growth Algorithm which aims at removing the bottlenecks of the Apriori-Algorithm in producing and testing candidate set. A unique, compact data structure, called frequent pattern tree or FP-tree therefore grounded on this structure an FP-tree-centered pattern fragment growth manner was developed. FP-growth uses a mixture of the vertical and horizontal database arrangement to store the database in main memory.

As an alternative of storing the cover for every item in the database, it stores the genuine transactions from the database in a tree structure and each item has a linked list passing through all transactions contain that item. This novel data structure is meant by FP-tree. Basically, all transactions are stored in tree-like data structure. The FP-tree is built in the following stages:

I. First the scanning of the transaction database DB is done once. Then, assemble the set of frequent items F and their supports. F is sorted in support of downward order as L, the list of frequent items.
II. Construct the root of an FP-tree, T, and tag it as "root".

Table 1. Comparative Analysis of different FIM Techniques

Author's Name	Technique	Characteristics	Dataset	Tool/ Platform	Parameter	Benefits	Limitation
Agarwal et al.(1994)	Apriori	Level wise search, Monotonicity property and Easy to implement	Synthetic Transaction	Java	Number of transactions Number of items	Generates frequent itemsets and association rules	Scalability
Zaki et al. (1997)	Eclat	Depth First Search, Works on vertical database intersection of tid_list	Synthetic and real dataset	Java	Minimum Support	Enhances locality and requires few scans to database	Degraded performance with larger number of transactions
Zaki et al. (2003)	dEclat	Uses vertical databases and diffsets over tidset	Mushroom	Hadoop	Minimum support Execution Time	Significant performance improvements	For sparse database diffsets loses its advantage over tidset
Han et al.(2000)	FP-Growth	Recursive approach, Employs FP-tree data structure	Connect Accident	RedHat, Linux C++	Runtime , Memory Consumption, Scalability	Focused search of smaller databases	Poor Performance
Linet al.(2012)	SPC,FPC,DPC	SPC-Simple implementation of Apriori on Map Reduce framework, FPC-single Map Reduce phase with merging of fixed passes and	Accident dataset, T1014D100K, Chess, Mushroom Retail	Hadoop with 7 map task and 1 reduce task	Confusion Matrix Size up Minimum Support	FPC and DPC provide efficient implementation of Apriori on	SPC increasing scheduling and waiting overhead and FPC may get overloaded

		DPC-Dynamically combine passes	Market			Map Reduce framework and reduce scheduling,invocation, increasing node utilization, workload balancing.	in case of large number of candidates.
Li et al. (2012)	PApriori	Sizeup, Speedup and Scaleup are used for performance evaluation.	Retail Chess	Hadoop Map Reduce	Minimum Support	Efficient and Good performance for large database	User need to give number of reducers
Hammou d. (2011)	MRApriori	Single scan of data in original format and Hybrid data structure, both horizontal and vertical	Retail	Hadoop Map Reduce	No of mapper No of Reducer	Efficient and Good performance for large database	No significant reduction in processing time.
Li et al. (2008)	Parallel FP Growth	Parallel version –FP – Growth ,Independent mining of FP-tree and grouping of items	URLs,Tags Transaction	Hadoop Map Reduce	Scalability, Run Time	Linear scalability	Not efficient in terms of memory and speed.
Zhou et al. (2010)	Balanced FP-Growth	Improvement in FP-Growth and uses frequencies of frequent items to balance the groups of PFP	Retail	Hadoop Map Reduce	No of Transaction	Faster execution using singletons with balanced distribution	Search Space partition using single item is not most efficient way.
Riondato et al.(2012)	PARMA	Use random sampling method	Mushroom	Hadoop Map Reduce	No of Transaction Speedup Runtime Accuracy	Minimizes data replication, Scaling linearly, Runs faster	Finds approximate collection of frequent itemsets.
Moens et al. (2015)	Dist-Eclat	Distributed version of Eclat	Mushroom Retail	Hadoop Map Reduce	Minimum Support	Speed	Scalability When Data size increases it does not work.

IV. CONCLUSION

A comparison framework has developed to allow the flexible comparison of existing and new frequent itemset mining algorithms that conform to the defined algorithm interface. Using this framework this paper presented the comparative performance study of iterative algorithms for FIM algorithms. In this work, an in-depth analysis of few algorithms is done which made a significant contribution to the search of improving the efficiency of frequent itemset mining. The developed framework can be used for comparing the other algorithms, which does not use candidate set generation to discover frequent patterns and can also lead to several ideas for optimizations, which could improve the performance of other algorithms**.**

References

1. **Gole, S., & Tidke, B. (2015, January).** Frequent Itemset Mining for Big Data in social media using ClustBigFIM algorithm. In Pervasive Computing (ICPC), 2015 International Conference on (pp. 1-6). IEEE.
2. **Hanirex, D. K., & Kaliyamurthie, K. P. (2014).** Mining frequent itemsets using genetic algorithm. Middle-East Journal of Scientific Research, 19(6), 807-810.
3. **Huang, D., Song, Y., Routray, R., & Qin, F. (2015, March).** Smart cache: An optimized mapreduce implementation of frequent itemset mining. In Cloud Engineering (IC2E), 2015 IEEE International Conference on (pp. 16-25). IEEE.
4. **Kovacs, F., & Illés, J. (2013, July).** Frequent itemset mining on hadoop. InComputational Cybernetics (ICCC), 2013 IEEE 9th International Conference on (pp. 241-245). IEEE.
5. **Labade, S., & Kini, S. N.** A Survey Paper on Frequent Itemset Mining Methods and Techniques.
6. **Lin, M. Y., Lee, P. Y., & Hsueh, S. C. (2012, February).** Apriori-based frequent itemset mining algorithms on MapReduce. In Proceedings of the 6th international conference on ubiquitous information management and communication (p. 76). ACM.
7. **Li, N., Zeng, L., He, Q., & Shi, Z. (2012, August).** Parallel implementation of apriori algorithm based on MapReduce. In Software Engineering, Artificial Intelligence, Networking and Parallel & Distributed Computing (SNPD), 2012 13th ACIS International Conference on (pp. 236-241). IEEE.
8. **Moens, S., Aksehirli, E., & Goethals, B. (2013, October).** Frequent itemset mining for big data. In Big Data, 2013 IEEE International Conference on (pp. 111-118). IEEE.
9. **Saabith, A. S., Sundararajan, E. A., & Bakar, A. A. (2016**). Parallel implementation of Apriori algorithms on the Hadoop-MapReduce platform-An evaluation of literature. Journal of Theoretical and Applied Information Technology, 85(3), 321-351.
10. **Zhang, Z., Ji, G., & Tang, M. (2013, December).** Mreclat: an algorithm for parallel mining frequent itemsets. In Advanced Cloud and Big Data (CBD), 2013 International Conference on (pp. 177-180). IEEE.
11. **Zhou, L., Zhong, Z., Chang, J., Li, J., Huang, J. Z., & Feng, S. (2010, November**). Balanced parallel FP-growth with mapreduce. In Information Computing and Telecommunications (YC-ICT), 2010 IEEE Youth Conference on (pp. 243-246). IEEE.

Author Biography

Dr.S.Kanmani

Professor S.Kanmani obtained her B.E and M.E degrees from Bharathiar University in computer science and engineering discipline in the year 1990(May) and 1991(December) with University rank. She has served as a lecturer in the Department of CSE, Annamalai University for a short period and joined as a lecturer in the Department of CSE, Pondicherry engineering college in the year 1992. Since then she was in various positions in the same institute and currently she is professor in the Department of Information Technology. She has completed her Ph.D in college of engineering, Guindy Anna University Chennai in the year 2006 in the area of software metrics.

She has produced 11Ph.D scholars and 3scholars are currently carrying out research under her guidance. She has published around 150 papers in the international conferences and reputed journals. She is a reviewer for many international journal publications.

M.Nithya

She is pursuing her M.Tech degree in the Department of Information Technology from Pondicherry Engineering College.