Research Journal of Computer and Information Technology Sciences _____ISSN 2320 – 6527
Vol. **12(1),** 1-4, June **(2024)** *Res. J. Computer and IT Sci.*

Mining Frequent Itemsets Based on Tree Structure from Transactional Dataset

Tusharkumar S. Patel[1*] and Harshad B. Bhadka[2]
[1]Faculty of Technology & Engineering, C. U. Shah University, Wadhwan City, Gujarat, India
[2]RDI Centre, C. U. Shah University, Wadhwan City, Gujarat, India
tusharit85@gmail.com

Available online at: www.isca.in, www.isca.me
Received 26rd July 2023, revised 10th February 2024, accepted 19th April 2024

Abstract

Research on mining frequent patterns is one of the emerging task in knowledge discovery. Many researchers have been studied efficient frequent itemset finding methods. All the previously available algorithms for mining frequent itemsets from transactional dataset are not efficient. The efficiency of algorithms is dependent on process of candidates generation, the structure which is used for storing generated candidates and the implementation. In this study, we propose a newly discovered Frequent Itemset Tree (FI-Tree) data structure. It is used for stowing frequent itemsets and its associated Transaction ID sets. In several data characteristics, MFIBT have a unique feature is that it has runs speedy. Our study shows that a MFIBT has better performs in terms of run time and memory consumption on transactional dataset.

Keywords: Itemsets, Data Mining, Frequent Pattern Mining.

Introduction

Data mining is a main area of research to find hidden knowledge or information from datasets is called knowledge discovery or data mining. The hidden information within datasets are used to find association relationships among sets of items which is used to discovery of association rules. The association rules are used in store layout, financial forecast, cross marketing, medical diagnosis, marketing policies, decision making and many other real life applications.

In data mining, frequent pattern finding is one of the most explored field. Frequent pattern finding have mainly three directions i: frequent itemset mining ii: sequential pattern mining and iii: sub-structure mining. Frequent itemsets are items exists in a dataset with itemset support is greater than a user-specified threshold.

Frequent itemsets are used in different knowledge mining problems that find most important itemsets from datasets like associated rules, classifiers, sequences, correlations, clusters etc. Out of which associated rules is crucial research problem. And frequent itemset finding is a sub-problem of association rules.

We newly discovered a Frequent Itemset Tree (FI-Tree) data structure. It is used for stowing frequent itemsets and its associated Transaction ID sets. In several data characteristics, MFIBT have a unique feature is that it has runs speedy. It has strong performance in different kinds of dataset, better than the pre-existing available methods in various parameters and its scalability is high for finding from transactional dataset.

In this paper, next section presents frequent itemset finding related work. After that, list out the steps of MFIBT algorithm for frequent itemset finding. Then explain the experimental results of algorithms for mining frequent itemsets. Finally, last section presents the conclusion.

Related work: In 1993, Agrawal et al. was first proposed method for frequent itemset mining from transactional dataset in the shape of mining associated rules. It examines customers shopping styles by mining association rules between various set of frequent items. For example, if customers are purchasing digital camera, out of which how many percentage of customers are interested to buy memory card on the one time to the store in market? This kind of knowledge can used to improve trades by proper decision making, cross marketing and store layout[1].

There are thousands of research articles exist on frequent itemsets finding methods with several types of improvements and implementations. Scalability of algorithms to handles variety of different datasets and different mining problems in a diversity of applications.

The Apriori algorithm is a level wise searching method and requires number of dataset scans on horizontal layout based transactional dataset[2]. Eclat algorithm works on vertical layout based transactional dataset, which is better in an execution time but requires large space of main memory[3]. FP-Growth algorithm works on projected layout based transactional dataset, which is better than all discussed above algorithms because no candidate generates but links store in main memory[4].

Also extensive studies on the Split and Merge (SaM) algorithm for frequent itemset generation[5].

The total number of transactions that exists the itemset is called support count of the itemset. An itemset contains set of items. A k-itemset is a itemset have k items and it is commonly denoted by L_k. Apriori and FP-Tree (AFPT) algorithm is a mixer of Apriori and FP-Growth methods. Its working mechanism is faster as compared to Apriori and FP-Growth methods[6].

Methodology

We provide a concise procedure of the Mining Frequent Itemset Based on Tree structure (MFIBT) for generating frequent itemsets discussed below:

A newly discovered data structure is Frequent Itemset Tree (FI-Tree). It is used for stowing frequent itemsets and its associated Transaction ID sets.

Now, frequent itemset mining MFIBT algorithm steps are given below: i: Suppose no more memory required, frequent 1-itemsets, transaction dataset and also main memory is exists for producing 2-itemsets candidate based on frequent 1-itemset. Scan dataset one time and produce frequent 1-itemsets with the parallel generate transaction sets, which exists the Itemset. ii: Produce 2-itemets candidate based on frequent 1-itemset only. iii: Resulted 2-itemsets candidate node count is less than user specified minimum threshold with the help of FI-Tree data structure, it will be eliminated. Now at the second level, FI-Tree contains only frequent 2-itemsets. iv: Similarly, approve the itemset by scan the frequent itemsets and its associated Transaction ID sets for each frequent 3, 4… n–itemset.

A frequent itemset mining based example on the MFIBT in Figure-1. In this example, the dataset have a six transactions means |D|=6.

Results and Discussion

To compare performance of algorithms, experiments were conducted on Intel® corei3™ CPU, 2.13 GHz, and 3 GB of RAM computer with Microsoft Windows 7 Home Basic Version 2009 Service Pack 1 operating system. All algorithms were coded using JAVA language. The Retail dataset was donated by Tom Brijs and contains the (anonymized) retail market basket data from an anonymous Belgian retail store and obtained from the FIMI repository[7]. The Adult dataset extraction was done by Barry Becker from the 1994 Census database and obtained from the UCI machine learning repository[8]. The Mushroom dataset was prepared by Roberto Bayardo from the UCI datasets and PUMSB and obtained from the FIMI repository[9]. The datasets and their properties are shown in Table-1.

Table-1: Datasets and Their Properties.

Datasets	Type	No. items	Average length	No. transactions	Size (KB)
Retail	Sparse	16470	10.3	88162	4156
Adult	Dense	134	14	48842	5082
Mushroom	Dense	119	23	8124	565

Table-2 to Table-7 in Figure-2 to Figure-7 respectively shows the comparison of the time and memory consumption are tested for the MFIBT, SaM, AFPT and Apriori algorithms with three datasets Retail, Adult and Mushroom and varying minimum support thresholds for discovering the frequent itemsets.

Figure-2 display our test results for Retail dataset using the execution time vs. different minimum support. Also peak memory in Mbytes requirements shown in Figure-3. We can see that the MFIBT is around 2.1 times faster than Apriori, 1.8 times faster than SaM algorithm and 1.3 times faster than AFPT algorithm with minimal support at 5%.

Figure-4 display our test results for Adult dataset using the execution time vs. different minimum support. Also peak memory in Mbytes requirements shown in Figure-5. We can see that the MFIBT is around 1.6 times faster than Apriori, 1.3 times faster than AFPT algorithm and 1.2 times faster than SaM algorithm with minimal support at 30%.

Figure-6 display our test results for Mushroom dataset using the execution time vs. different minimum support. Also peak memory in Mbytes requirements shown in Figure-7. We can see that the MFIBT is around 1.6 times faster than Apriori, 1.2 times faster than SaM algorithm and 1.2 times faster than AFPT algorithm with minimal support at 30%.

Table-2: Retail dataset total run time.

Support (in %)	Total run time in second			
	MFIBT	SaM	AFPT	Apriori
25	2.037	2.652	2.308	2.827
20	2.044	3.082	2.372	3.281
15	2.075	3.158	2.402	3.822
10	2.106	3.742	2.466	4.135
5	2.215	3.898	2.808	4.65

Table-3: Peak memory required for Retail dataset.

Support (in %)	Peak Memory in Mbytes			
	MFIBT	SaM	AFPT	Apriori
25	60.818	62.681	61.202	58.886
20	61.246	63.952	65.258	60.354
15	61.52	65.025	69.195	62.754
10	61.676	67.53	74.376	63.493
5	61.883	68.958	78.453	64.736

Table-4: Adult dataset total run time.

Support (in %)	Total run time in second			
	MFIBT	SaM	AFPT	Apriori
70	1.01	1.7	2.28	2.96
60	1.99	2.69	2.91	3.863
50	3.56	4.51	4.74	5.464
40	5.69	6.72	7.6	9.616
30	8.12	9.85	10.92	13.28

Table-5: Peak memory required for Adult dataset.

Support (in %)	Peak Memory in Mbytes			
	MFIBT	SaM	AFPT	Apriori
70	81.36	96.9	171.9	98.4
60	91.5	142.1	190.8	106.5
50	100.21	209.25	251.44	124.05
40	189.88	224.9	280.81	194.81
30	234.26	246.2	378.36	245.33

Table-6: Mushroom dataset total run time.

Support (in %)	Total run time in second			
	MFIBT	SaM	AFPT	Apriori
70	0.375	0.577	0.624	0.81
60	0.421	0.562	0.646	0.966
50	0.53	0.734	0.862	1.366
40	0.858	1.638	2.076	2.404
30	1.888	1.954	2.184	2.951

Table-7: Peak memory required for Mushroom dataset.

Support (in %)	Peak Memory in Mbytes			
	MFIBT	SaM	AFPT	Apriori
70	22.23	26.475	57.3	73.67
60	25	38.825	63.6	75.6
50	27.38	68.7	69.75	78.5
40	51.88	76.725	84.3	88.55
30	111.42	130.7	115.4	102.13

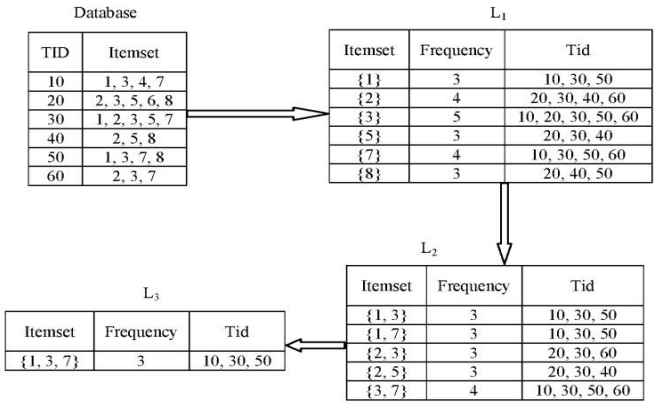

Figure-1: Mining frequent itemsets based on the MFIBT.

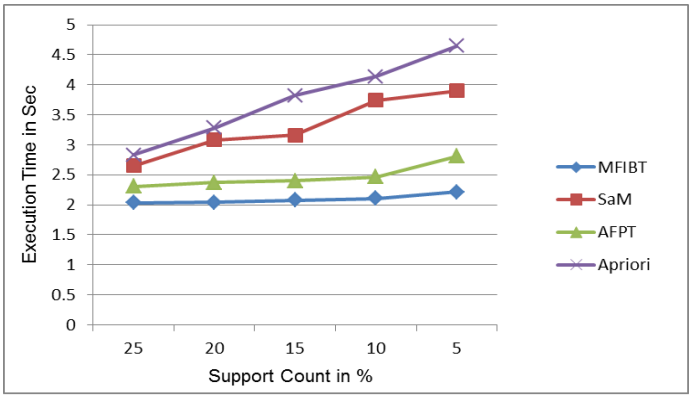

Figure-2: Total execution time of Retail dataset.

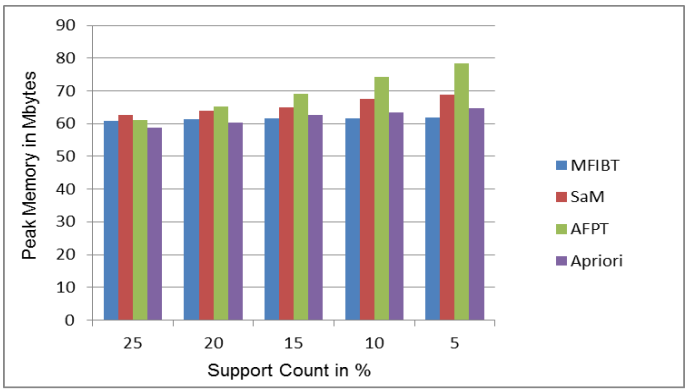

Figure-3: Peak memory required for Retail dataset.

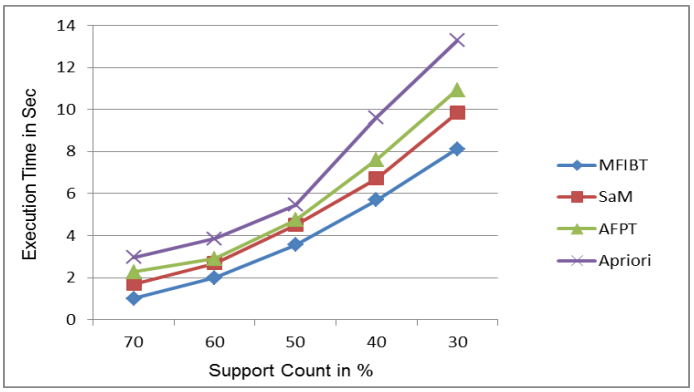

Figure-4: Total execution time of Adult dataset.

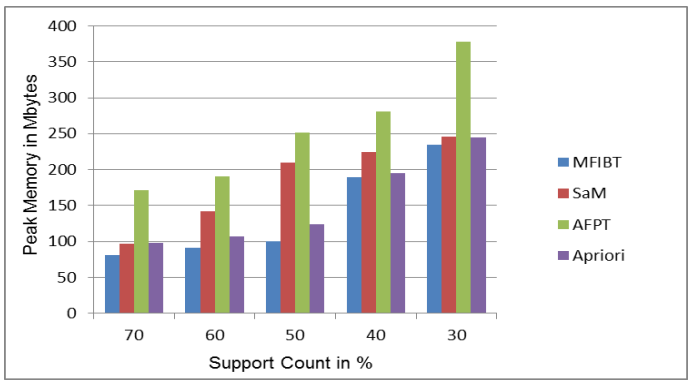

Figure-5: Peak memory required for Adult dataset.

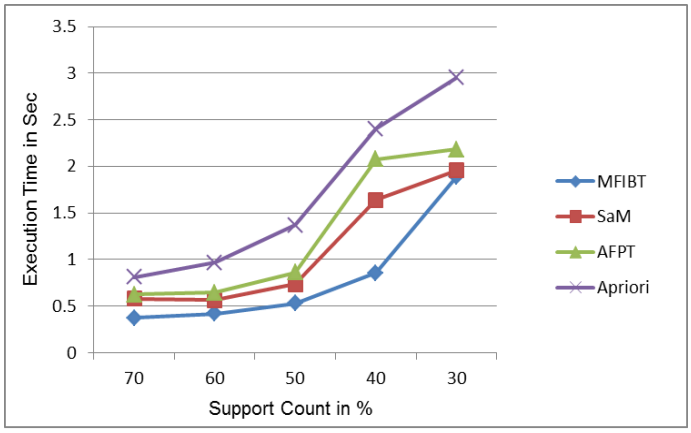

Figure-6: Total execution time of Mushroom dataset.

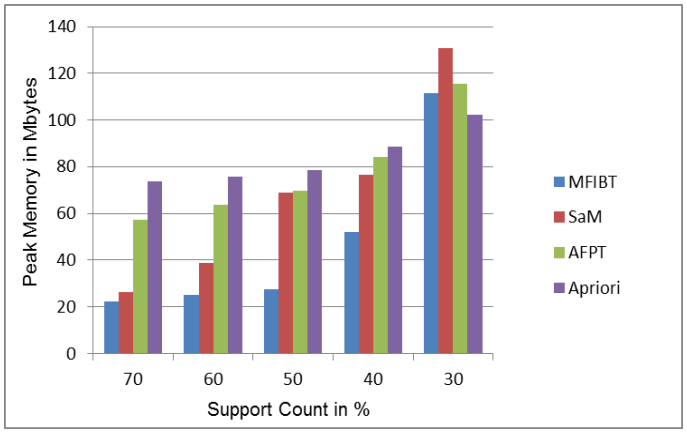

Figure-7: Peak memory required for Mushroom dataset.

Conclusion

In this paper, we conduct the experiments on MFIBT algorithm to find frequent itemsets using a newly discovered Frequent Itemset Tree (FI-Tree) data structure. It is used for stowing frequent itemsets and its associated Transaction ID sets. In several data characteristics, MFIBT have a unique feature is that it has runs speedy. The experiments includes run time and space consumption are tested for the MFIBT, SaM, AFPT and Apriori algorithms using transactional dataset and varying minimum support threshold. Changing the minimum support does not major affect the MFIBT algorithm in run time and space consumption.

The node construction of tree structure and number of recursive steps of mining do not required rescanning of the database, thereby reducing the total execution time. The experimental results demonstrated that the MFIBT algorithm is better than other three algorithms in run time and space consumption in dense datasets. It is best executed in dense datasets while in sparse datasets, its performance towards other three algorithms is better but slightly decreases.

The simple conclusion can be made where the nature of datasets could be one of the contributing factor to the overall performance of algorithms in frequent itemset mining.

References

1. Agrawal, R., Imieliński, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. In Proceedings of the 1993 ACM SIGMOD international conference on Management of data, 207-216.

2. Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. In Proc. 20th int. conf. very large data bases, VLDB, Vol. 1215, 487-499.

3. Borgelt, C. (2003). Efficient implementations of apriori and eclat. In FIMI'03: Proceedings of the IEEE ICDM workshop on frequent itemset mining implementations (Vol. 90).

4. Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. *ACM sigmod record*, 29(2), 1-12.

5. Borgelt, C. (2010). Simple algorithms for frequent item set mining. In Advances in Machine Learning II: Dedicated to the Memory of Professor Ryszard S. Michalski (pp. 351-369). Berlin, Heidelberg: Springer Berlin Heidelberg.

6. Lan, Q., Zhang, D., & Wu, B. (2009). A new algorithm for frequent itemsets mining based on apriori and FP-Tree. In 2009 WRI Global Congress on Intelligent Systems (Vol. 2, pp. 360-364). IEEE.

7. Goethals, B. (2003). Frequent Itemset Mining Dataset Repository. http://fimi. cs. helsinki. fi/data/.

8. Asuncion, A., & Newman, D. (2007). UCI machine learning repository.

9. Bayardo, R. (2014). Frequent itemset mining dataset repository. UCI datasets and PUMSB.