Mining Closed Item sets using Partition Based Single Scan Algorithm

U. Mohan Srinivas, E. Srinivasa Reddy

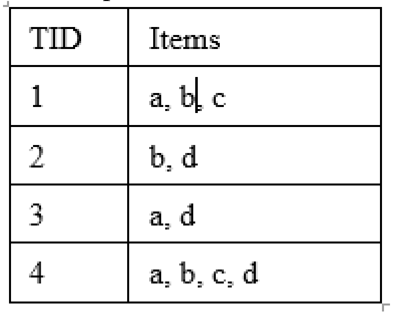
Abstract**:** ***Closed item sets are frequent itemsets that uniquely determines the exact frequency of frequent item sets. Closed Item sets reduces the massive output to a smaller magnitude without redundancy. In this paper, we present PSS-MCI, an efficient candidate generate based approach for mining all closed itemsets. It enumerates closed item sets using hash tree, candidate generation, super-set and sub-set checking. It uses partitioned based strategy to avoid unnecessary computation for the itemsets which are not useful. Using an efficient algorithm, it determines all closed item sets from a single scan over the database. However, several unnecessary item sets are being hashed in the buckets. To overcome the limitations, heuristics are enclosed with algorithm PSS-MCI. Empirical evaluation and results show that the PSS-MCI outperforms all candidate generate and other approaches. Further, PSS-MCI explores all closed item sets.***

Index Terms**:** ***data mining, frequent itemsets, closed itemset, minimum support.***

I. INTRODUCTION

Nowadays, huge amounts of data are collected from various resources and available to everyone. Due to the complexity of data and the need of various applications, the extraction of interested information such collection is an active research area. Data mining is an active research are in retrieving hidden, valuable and unknown information from a large collection of data or database. In that, Frequent Itemset Mining (FIM) is one of the popular data mining technique that aims at extracting itemsets that are highly correlated as hidden knowledge from a transactional database. FIM is formally formulated as, from a given list of transactions, minimum threshold, find all the itemsets whose occurrence is at least minimum support count. FIM goal is to find the Frequent Itemsets (FI), a set of items whose occurrence is greater than the minimum support of all transactions. One of the basic applications is market basket analysis [3], where each transaction corresponds to a set of products purchased by a customer. To analyze the purchase behavior, find a set of products which occurs together in a minimum threshold percentage of transactions. It can be mapped to many real scenarios of applications, it is mapped to other topics Frequent Episode Mining, Sequential Pattern Mining, Classification and Clustering. In FIM, several approaches have been prosed for FIM [4], classified into two groups, they are CGAT (candidate generation and test) and other is without candidate generation that is FP-Growth [13]. The reputed algorithm under first category is Apriori [2, 3], which runs on the heuristic Apriori and anti-monotonic property. The second category is based on tree concept rather than candidates, where the entire data base is represented in a tree and do mine tree recursively to extract all frequent itemsets. However, the number of FI's that are extracted from large databases can be huge which requires huge storage area and more computations. For example, Table 1 is recorded with a list of 4 transactions, consider the minimum support min-sup=50% (count = 2). FIM { a:3, b:3, c:2, ab:2, ac:2, ad:2, bc:2, bd:2, abc:2}. As per the definition of closed itemset, it is observed that {c:2, ab:2, ac:2, bc:2} can be determined from {abc:2}. Hence {c:2, ab:2, ac:2, bc:2} are considered as redundant.

Table 1: Sample Transactional Database



TID	Items
1	a, b, c
2	b, d
3	a, d
4	a, b, c, d

As a result, several condensed representations for FI's have been proposed to reduce the size of FI's without losing knowledge [8]. The very next alternation method was Maximal Itemsets, a set of itemsets whose support reaches the threshold and doesn't have any superset. It has shown very impact on the size of FI's. MaxClique, Mafia [6], Pincer search [16], Maxminer [Bayardo 98], Depth project [3], Mafia [6], GenMax [12] and FPMax [12]. All the above algorithms are able to extract all the maximal itemsets. However, multiple scans of database was needed when the main memory size was small and too many possible itemsets were generated at each pass. However, extracting frequent information with exact support is not achievable. Further, it has been investigated, the result with the term Closed Itemsets CI. CI is a set of itemsets which doesn't have any supersets with the same support. The research including top-down approaches [7, 5, 20], Bottom-up approaches and combination of both is Pincer search [16, 17]..

Revised Manuscript Received on 30 July 2019.
* Correspondence Author

U. Mohan Srinivas, Research Scholar, CSE, ANUCET, Guntur, India.
E. Srinivasa Reddy, Prof. and Dean, CSE, ANU, UCET, Guntur, India .

The above approaches have shown the output contains all the frequent itemsets. However, multiple scans of database was needed when the main memory size was small and too many possible itemsets were generated at each pass.

Contributions:

In this paper, we propose a Novel approach called Partition Based Single Scan Approach (PSS-MCI) for Mining Closed Itemsets. Hash Table is used to capture the Possible Frequent Itemsets, Frequent Itemsets and Closed Itemsets which are generated for each transaction. Hash Table helps us to maintain unique itemsets and cumulating itemsets support. Function isClosed is proposed to perform super-set and sub-set checking. Heuristics on Partition based approach is proposed to avoid unnecessary computation on infrequent itemsets. Including all the above, PSS-MCI able to derive all closed frequent itemsets with a single scan. The rest of the paper is organized as follows. The very next section review the FIM, Closed Itemsets algorithms. In the next section, the proposed approach and its description is presented. The result analysis of the proposed method is presented in next section. Conclusion is presented in the last section.

II. RELATED WORK

FIM is the fundamental technique for the most of the data mining tasks, such as Mining Correlation among the items of database, Association Rules [3, 9, 10, 11, 32], Classification [17], approximation [11]. Several algorithms have been proposed to improve the performance of FIM, such are parallel and single scan algorithms [29]. The investigation on FIM has been carried out and resulted with many algorithms, such as Maximal [2, 6, 12, 16, 32], Minimal, Closed [8], Generators, and Sequential Patterns [1]. Mining Closed Itemsets become popular since it doesn't exhibit redundancy and no loss of information. We classify FIM algorithms into three categories:

(i)Candidate Generate and Test: This category contains algorithms exhibits the property Apriori : An itemset X is frequent, if and only if all of its subsets are frequent. This property explores frequent patterns with multiple scans over database. But this idea lead to many developments such as A-CLOSE[20], ECLAT[30, 31], TITANIC [23].

(ii) Tree Based –Divide and Conquer: This category contains algorithms based on tree FP-Tree and FP-Growth [13]. They maintains entire information in FP-Tree and mine it in recursive manner to get all interested itemsets. CLOSET [21], CLOSET+[28], FPCLOSE[12] and AFONT[17]. CLOSET mines closed patterns by visiting tree in depth-first manner and computing closed closure. To improve the performance of CLOSET whose performance is an adequate for sparse and dense, CLOSET+ is introduced upward checking to avoid duplication computation. It needs to maintain additional storage for performing intersection to perform updward checking. To overcome, FP-CLOSE[12] is proposed and popular as the best approach for mining closed itemsets when the entire data fits into the memory.

(iii) Mixed Approaches: To overcome the limitations of the above two approaches, mixed category contains the approaches that are adopted from candidate generation and tree based approaches. CHARM [31] is the known to be one of the popular because of its data structure ItemsetTidSet Tree. It does depth-first manner rather than splitting tree into multiple and do the same in each split. It does closed itemsets by doing on data structure. Other algorithms are CLOSEMINER[22], PGMINER[19], DCI-CLOSED [18], LCM[24, 25, 26], DBV-MINER[27] and FCP-MINER[15].

The above classification performance depends on the nature of the data. Hence it is difficult to pick the best approach for mining closed itemsets. Keeping in mind that having less memory and less scans on database, we propose a single scan approach that uses partition based approach and candidate generation concept, heuristics to avoid unnecessary candidates to derive all closed itemsets.

III. PARTITION BASED SINGLE SCAN APPROACH FOR MINING CLOSED ITEMSETS (PSS-MCI)

This section presents and discusses the proposed algorithm PSS-MCI, which uses Hash table, Candidate generate strategy and Heuristics to derive closed frequent itemsets. Closed Itemsets: Closed itemsets are the frequent itemsets that do not have any supersets with the same support. These kind of itemsets are also called as the lossless largest frequent itemsets. It can be represented as CI ⊆ L, where CI is the Closed itemsets and L is the set of frequent itemsets. When CI are subsets of L, and X is a frequent itemset, Y ⊃ X where Y is Closed, and all its subsets are also frequent. From the above two points, it can say that CI reduces the search space.

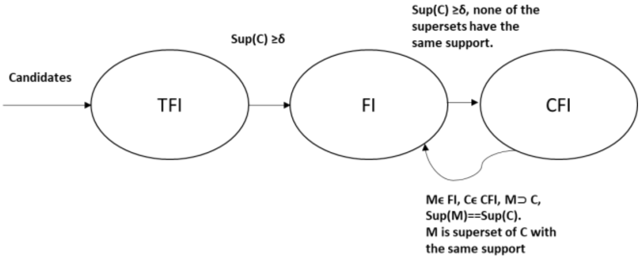

Sup(C) ≥δ, none of the supersets have the same support.
Sup(C) ≥δ
Candidates
TFI
FI
CFI
M∊ FI, C∊ CFI, M⊃ C, Sup(M)==Sup(C). M is superset of C with the same support

Fig 1: General Approach –Closed Itemset Mining

3.1. *Description*:

PSS-MCI aims at reducing multiple scans over the database and mine all possible to derive redundant less frequent itemsets Closed Itemsets. The basic idea of this approach is similar to the partition based approaches which is to divide the transactions into equal partitions M where the size of each partition is same as half of the minimum support, and generate all the candidates for each transactions. It assumes that the generated itemsets are PFI (Possible Frequent Itemsets). If I is a generated itemset that has already indexed in hash table, then it increments the support by one. Otherwise, it creates a new entry with the itemset name in hash table initializes counter value with one. Itemsets moves from either PFI to FI (Frequent Itemsets), FI to CI (Closed Itemsets) or vice versa. This process is repeated for all the partitions. The main contribution of this work is to avoid the testing infrequent itemsets which are not going to become closed. After visiting half of the partitions, if I is newly generated itemset and not indexed in the hash table then it is discarded, such kind of itemsets are not stored in hash table.

Retrieval Number: A1920058119/19©BEIESP
DOI: 10.35940/ijrte.A1920.078219
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Hence the computation for such kind of unnecessary itemsets is achieved and no information is lost in the mining process. In addition to that, it also performs superset and subset checking to avoid redundant itemsets that are maintained in a list which are frequent but not closed. Such itemsets are removed from hash. Finally it gives only closed itemsets in hash table. PSS-MCI is complete, because all closed frequent itemsets are derived directly from the candidate

Item sets which are generated directly from the given TDB, whose support is ≥ minsup and doesn't have any supersets with the same support. After visiting minsup of TDB, if the newly generated itemset is not indexed in the hash table, there is no chance of getting the frequency of minsup. Hence it is complete. Algorithm 1 describes the step by step activities of PSS-MCI. Procedure PSS-MCI for mining closed itemsets is presented below. As a first step, It takes transactional database TDB as input, and δ (minsup) given by the user and it divides TDB into M partitions which are equally size. The size of each partition is the half of δ of TDB. In second step, it generates all the possible candidate itemsets for each transaction. In step 3, it uses data structure hash table to store the candidate itemsets in the form of PFI, FI and CI with their support value w.r.t the conditions mentioned in algorithm. Candidate itemset X ⊂ PFI moves from PFI to FI or CI, if Sup(X)≥δ, and moves from CI to FI if it finds superset with the same support. After visiting half of the partitions, the generated itemset is new to the hash table then it is discarded, if it is not going to be maximal, and if it is not going to be frequent. In fourth step, all the itemsets whose support is less than δ, are to be deleted from hash table. At the end, it returns all the itemsets which are associated with CI.

Procedure: PSS-MCI

Input: TDB, δ- User Minimum Threshold
Output: L_C: List of Closed frequent itemsets

```
Symbols: FI – Frequent Itemsets, CFI- Closed Frequent Itemsets PFI- Possible Frequent Itemsets
Step 1: Partition TDB into M equal partitions whose size is same as the half of the min-sup.
Step 2: Each Partition $M_i$, for each transaction $T_i \in M_i$, generate all the possible candidate
            itemsets to Cand.
Step 3: for each C of Cand, consider as PFI  do the following steps
        if C is not indexed already in hash,
        then
            $PFI \leftarrow PFI \cup C$
            $C.sup \leftarrow C.sup + 1$
        else                                // it is already indexed in hash
        increment C.Sup by 1
        if $Sup(C) \geq \delta$
            if isClosed (C)
                Update FI w.r.t C
            Else
                If $C \in PFI$
                    Then $FI \leftarrow FI \cup C$
        else $PFI \leftarrow PFI \cup C$
Repeat step 2 and 3 till all the transactions of all partitions are visited.
Step 4: Remove the indexes whose support value is Sup (h(t))<minsup .
Step 5: apply Superset checking, if it is required.
```

Heuristic 1:

Let say c be a newly generated candidate itemset, if is not indexed in hash and Ti > $(\frac{|TDB|}{2})$, then itemset c can be ignored, since it will not become frequent.

Heuristic 2: look forward:

Let assume c be a newly generated candidate itemset, Tk is the current transaction, the itemset c is not considered if the support of c is

$$Sup(hash(c) + (|TDB| - k) < minsup.$$

Heuristic 3:

If c is infrequent then all of its super sets are also infrequent. As per the Anti-monotonic property described in [2,3], the above property states true.

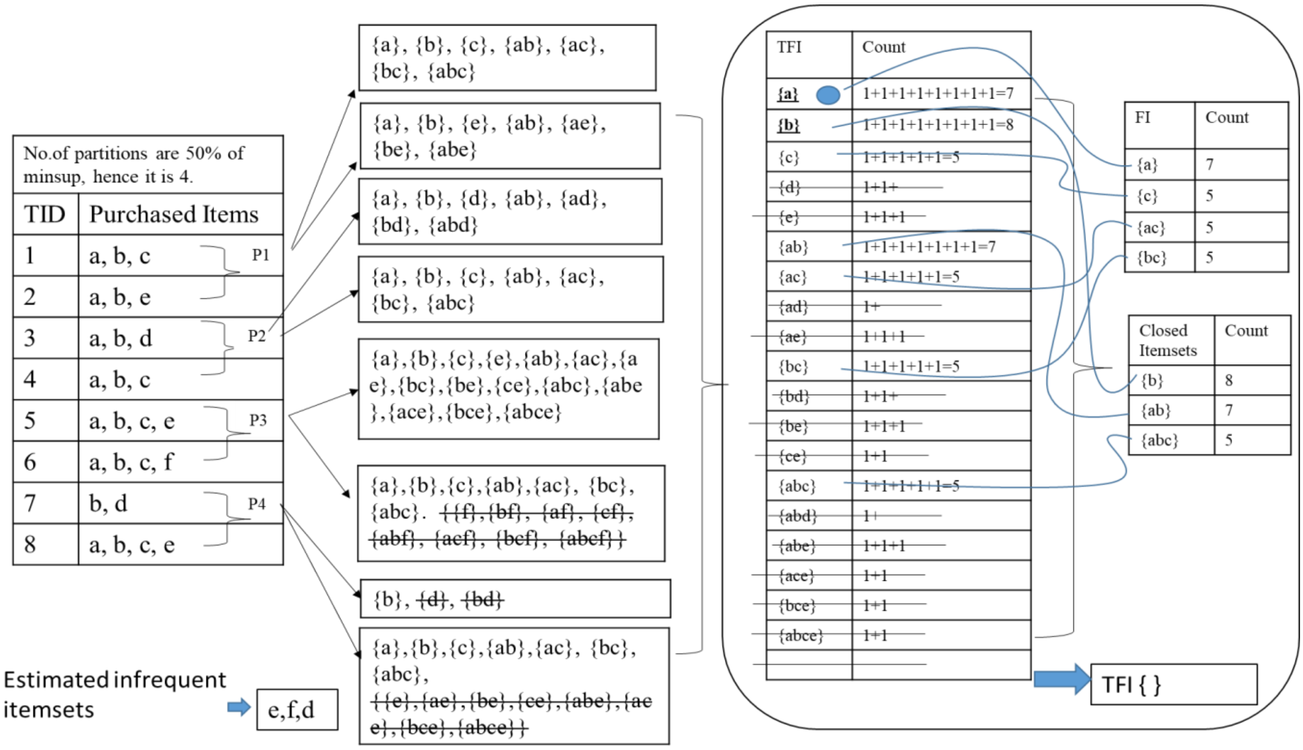
Table 2: Illustration of Transactional Database

TID	Purchased Items
1	a, b, c
2	a, b, e
3	a, b, d
4	a, b, c
5	a, b, c, e
6	a, b, c, f
7	b, d
8	a, b, c, e

3.2. Illustration:

Table 2 is considered for illustrating PSS-MCI. Figure2 shows the pictorial representation of PSS-MCI algorithm execution for the TDB of Table 1 with minsup=50% (0.5 or 4). It starts by partitioning TDB into 4 partitions, and it can be seen that the partitions are P1 {{a,b,c}, {a,b,e}}, P2 {{a,b,d} {a,b,c} }, P3 {{a,b,c,e} {a,b,c,f} } and P4 {{b,d} {a,b,c,e}.

Step 2: the possible combinations from transaction T1 {a, b, c} are {a}, {b}, {c}, {ab}, {ac}, {bc} and {abc}. Initially, the hash table h is empty, then all these itemsets are considered as PFI and stored into the h with the support 1. Same procedure is repeated for the rest of the transactions. During TID4, itemset {a} gets minimum support 4, and it moves to CI since FI and CI are empty. Itemset {ab} gets the same support. Hence {a} and {b} becomes frequent but not closed. The Same procedure is repeated until half of the partitions are visited. For a transaction TID6 and second of P3 is {a,b,c,f}, and the possible itemsets are { {a}, {b}, {c}, {f} {ab},{ac},{af},{bc},{bf},{cf}, {abc},{abf}, {bcf}, {abcf}}. The estimated support of {f} is not going to be frequent, hence {f} and its combinations are discarded. The same procedure is repeated for the remaining transactions. And the final result, closed itemsets are {{b}, {ab} and {abc}}.



No.of partitions are 50% of minsup, hence it is 4.		
TID	Purchased Items	
1	a, b, c	P1
2	a, b, e	
3	a, b, d	P2
4	a, b, c	
5	a, b, c, e	P3
6	a, b, c, f	
7	b, d	P4
8	a, b, c, e	

TFI	Count
{a}	1+1+1+1+1+1+1=7
{b}	1+1+1+1+1+1+1+1=8
{c}	1+1+1+1+1=5
~~{d}~~	~~1+1+~~
~~{e}~~	~~1+1+1~~
{ab}	1+1+1+1+1+1+1=7
{ac}	1+1+1+1+1=5
~~{ad}~~	~~1+~~
~~{ae}~~	~~1+1+1~~
{bc}	1+1+1+1+1=5
~~{bd}~~	~~1+1+~~
~~{be}~~	~~1+1+1~~
~~{ce}~~	~~1+1~~
{abc}	1+1+1+1+1=5
~~{abd}~~	~~1+~~
~~{abe}~~	~~1+1+1~~
~~{ace}~~	~~1+1~~
~~{bce}~~	~~1+1~~
~~{abce}~~	~~1+1~~

FI	Count
{a}	7
{c}	5
{ac}	5
{bc}	5

Closed Itemsets	Count
{b}	8
{ab}	7
{abc}	5

{a}, {b}, {c}, {ab}, {ac}, {bc}, {abc}
{a}, {b}, {e}, {ab}, {ae}, {be}, {abe}
{a}, {b}, {d}, {ab}, {ad}, {bd}, {abd}
{a}, {b}, {c}, {ab}, {ac}, {bc}, {abc}
{a},{b},{c},{e},{ab},{ac},{ae},{bc},{be},{ce},{abc},{abe},{ace},{bce},{abce}
{a},{b},{c},{ab},{ac}, {bc}, {abc}. ~~{{f},{bf}, {af}, {cf}, {abf}, {acf}, {bcf}, {abcf}}~~
{b}, ~~{d}~~, ~~{bd}~~
{a},{b},{c},{ab},{ac}, {bc}, {abc}, ~~{{e},{ae},{be},{ce},{abe},{ace},{bce},{abce}}~~
Estimated infrequent itemsets
e,f,d
TFI { }

Figure 2: PSS-CIM illustration of Table1

3.3. Result Analysis:

Several experiments have been conducted to show the performance of PSS-MCI with the standard data sets used in [14]. The first dataset is recorded with transactions of 2178, number of items from 8 to 16, and the average of items per transaction is between 8 and 16. The second data set with medium sized database size of 4 with transactions size varies from 59000 to 100000, number of items from 500 to 16000, average items is from 2 to 10. Third dataset is a large database BMPPOS, which is recorded with 500000 transactions and above 1660 items with average items 2.5. All the experiments are carried with the configurations of Intel I3 processor, 4GB RAM, and implemented in JAVA. Table 3 is recorded with runtime comparison of PSS-MIM, Apriori, Maxminer and MaxEclat using standard datasets. It is observed that ECLAT approach is performing better than other approaches. However for medium and large database, PSS-MCI outperforms Apriori more than twice and other approaches. In other way, we can say that PSS-MCI is efficient than apriori and others when the database is dense.

Table 3 Run time comparison of PSS-MIM in seconds

Data set	Apriori-closed	ECLAT	TITANIC	PSS-MCI
Bolts	7	5	6	5
Sleep	11	8	9	8
Pollution	30	20	22	16
Basket ball	22	18	20	16
Quake	35	30	30	27
BMS-Web View-1	1400	800	900	220
BMS-Web View-2	5420	3200	3500	1700
Retail	5840	4000	4020	2800
Connect	3400	2500	2300	1450
BMP POS	12500	6500	6800	3150

IV. CONCLUSION

This paper has proposed an intelligent closed itemset mining algorithm. It extracts all closed itemsets with a single scan on database with a less number of candidate itemsets compared to naïve and other approaches. Hash table data structure was used to maintain all the possible itemsets that are generated for each transaction. Heuristics are also proposed to speed up the execution process. Experimental results shows that PSS-MCI outperforms other approaches for large and dense databases.

In further, it is evident that more heuristics can be imposed to improve the execution time and the utilization of search space.

REFERENCES

1. Agrawal, R. and Srikant, R. (1995) ‘Mining sequential patterns’, in ICDE, pp.3-14.
2. Agrawal, R., Aggarwal, C., and Prasad, V. (2000) ‘Depth first generation of long patterns’ in Proceedings of Seventh International Conference on Knowledge Discovery and Data Mining, pp. 108–118.
3. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: ACM SIGMOD Record, vol. 22, no. 2, pp. 207–216. ACM, June 1993
4. Borgelt, C.: Frequent itemset mining. Wiley Interdisc. Rev.: Data Min. Knowl. Discov. 2(6), 437–456 (2012).
5. Boulicaut, J.-F., Bykowski, A., & Rigotti, C. (2003). Free-sets: a condensed representation of boolean data for the approximation of frequency queries. Data Mining and Knowledge Discovery 7 (1), 5–22.
6. Burdick, D., Calimlim,M., and Gehrke, J. (2001) ‘MAFIA: A maximal frequent itemset algorithm for transactional databases’, in Proceedings of IEEE International Conference on Data Engineering, pp.443–452.
7. Bykowski, A., & Rigotti, C. (2001). A condensed representation to find frequent patterns. In: Proceedings of the Twentieth ACM SIGMODSIGACT-SIGART Symposium on Principles of Database Systems. PODS ’01. ACM, New York, NY, USA, pp. 267–273.
8. Cerf, L., Besson, J., Robardet, C., Boulicaut, J.F.: Closed patterns meet n-ary relations. ACM Trans. Knowl. Discov. Data (TKDD) 3(1), 3 (2009).
9. Djenouri, Y., Bendjoudi, A., Mehdi, M., Nouali-Taboudjemat, N., Habbas, Z.: GPU-based bees swarm optimization for association rules mining. J. Supercomput. 71(4), 1318–1344 (2015).
10. Djenouri, Y., Drias, H., Habbas, Z.: Bees swarm optimisation using multiple strategies for association rule mining. Int. J. Bio-Inspired Comput. 6(4), 239–249 (2014).
11. Gheraibia, Y., Moussaoui, A., Djenouri, Y., Kabir, S., Yin, P.Y.: Penguins search optimisation algorithm for association rules mining. CIT J. Comput. Inf. Technol.24 (2), 165–179 (2016).
12. Grahne, G., & Zhu, J. (2005). Fast algorithms for frequent itemset mining using FP-Trees. IEEE Transactions on Knowledge and Data Engineering 17 (10), 1347–1362.
13. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: ACM SIGMOD Record, vol. 29, no. 2, pp. 1–12. ACM, May 2000
14. Hegland, M.: The apriori algorithm tutorial. Math. Comput. imaging Sci. Inf. Process. 11, 209–262 (2005).
15. Király, A., Laiho, A., Abonyi, J., & Gyenesei, A. (2014). Novel techniques and an efficient algorithm for closed pattern mining. Expert Systems with Applications 41 (11), 5105 – 5114
16. Lin, D. and Kedem, Z.M. (2002) ‘Pincer-Search : An Efficient Algorithm for Discovering the Maximum Frequent Set’, in IEEE Transactions on Knowledge and Data Engineering. Vol 14, No. 3, pp.553 – 566.
17. Liu, G., Lu, H., Yu, J. X., 0011, W. W., & Xiao, X. (2003). AFOPT: An efficient implementation of pattern growth approach. In: FIMI. Vol. 90 of CEUR Workshop Proceedings. CEUR-WS.org
18. Lucchese, C., Orlando, S., & Perego, R. (2006). Fast and memory efficient mining of frequent closed itemsets. IEEE Transactions on Knowledge and Data Engineering 18 (1), 21–36.
19. Moonesinghe, H. D. K., Fodeh, S. J., & Tan, P.-N. (2006). Frequent closed itemset mining using prefix graphs with an efficient flow-based pruning strategy. In: International Conference on Data Mining. IEEE Computer Society, pp. 426–435.
20. Pasquier, N., Bastide, Y., Taouil, R., & Lakhal, L. (1999). Efficient mining of association rules using closed itemset lattices. Information Systems 24 (1), 25 – 46.
21. Pei, J., Han, J., & Mao, R. (2000). CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets. In: ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery. pp. 21–30.
22. Singh, N. G., Singh, S. R., & Mahanta, A. K. (2005). Closeminer: Discovering frequent closed itemsets using frequent closed tidsets. In: International Conference on Data Mining. IEEE Computer Society, pp. 633–636.
23. Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., & Lakhal, L. (2002). Computing iceberg concept lattices with titanic. Data & Knowledge Engineering 42 (2), 189–222.
24. Uno, T., Asai, T., Uchida, Y., & Arimura, H. (2004a). An efficient algorithm for enumerating closed patterns in transaction databases. In: Discovery Science. Vol. 3245 of Lecture Notes in Computer Science. Springer, pp. 16–31.
25. Uno, T., Kiyomi, M., & Arimura, H. (2004b). Lcm ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. In: FIMI. Vol. 126 of CEUR Workshop Proceedings.
26. Uno, T., Kiyomi, M., & Arimura, H. (2005). Lcm ver.3: Collaboration of array, bitmap and prefix tree for frequent itemset mining. In: Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations. ACM, New York, NY, USA, pp. 77–86.
27. Vo, B., Hong, T.-P., & Le, B. (2012). Dbv-miner: A dynamic bit-vector approach for fast mining frequent closed itemsets. Expert Systems with Applications 39 (8), 7196–7206
28. Wang, J., Han, J., & Pei, J. (2003). Closet+: searching for the best strategies for mining frequent closed itemsets. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge Discovery and Data Mining. ACM Press, New York, NY, USA, pp. 236–245.
29. Youcef D, Marco C, Djamel: SS-FIM: Single Scan for Frequent Itemsets Mining in Transactional Databases. PAKDD, part II, LNAI 10235, pp. 644-654, 2017.
30. Zaki, M. J., & Gouda, K. (2003). Fast vertical mining using diffsets. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge Discovery and Data Mining. pp. 326–335.
31. Zaki, M. J., & Hsiao, C.-J. (2005). Efficient algorithms for mining closed itemsets and their lattice structure. IEEE Transactions on Knowledge and Data Engineering 17 (4), 462–478.
32. Zaki, M.J., Parthasarathy, S., Ogihara, M., Li, W.: New algorithms for fast discovery of association rules. In: Third International Conference Knowledge Discovery and Data Mining (1997).

AUTHORS PROFILE

U. Mohan Srinivas completed B.Tech and M.Tech. in Computer Science and Engineering from JNTU, Kakinada, India in 2004. He is pursuing Ph.D. in CSE department at ANU with the guidance from Prof. E. Sreenivasa Reddy. He has professional membership from CSI and interest in Intelligent Data Mining, and Pattern classifications.

Dr. E. Sreenivasa Reddy obtained B.Tech then M.S. degree from BITS, M.Tech (CS) from Visveswaraiah Technological University, India in 2000 and Ph.D in Computer science from Acharya Nagarjuna Univeristy, India in 2008. At present Ph.D scholars from several universities are under his guidance. Several National, International papers are published in reputed Journals as well as presented in conferences too. Senior membership of IEEE honored him. His research interest are in Imaging Technologies, Biometrics and spread to over many areas.