Partition Based Single Scan Approach for Mining Maximal Itemsets

U. Mohan Srinivas, E. Srinivasa Reddy

Abstract: ***In this paper, Frequent Itemset mining (FIM) limitations and compact representation of FIM that is Maximal Itemsets explored for extracting unknown redundant less frequent itemsets from the transactional database. The candidate generation and support calculations are the major tasks in FIM. FIM challenges with the following limitations for low support threshold: (i) huge frequent itemsets are generated as output (ii) difficulty in taking decision among the frequent itemsets due to redundancy. (iii) To find the cumulative support of itemsets, database scan is required for each length. The first issue can be resolved using maximal itemsets that are frequent who doesn't have any superset is called as Maximal Itemset Mining (MIM). Maximal itemsets are useful in minimal key discovery kind of applications. Hence, we present a Single scan algorithm to address the above limitations. However, several unnecessary itemsets are being hashed in the buckets. To overcome the limitations, a partition-based approach is proposed. Empirical evaluation and results visualize that the PSS-MIM outperforms all candidate generate and other approaches.***

Index Terms: data mining, frequent itemsets, Maximal itemset.

I. INTRODUCTION

Data mining excels in retrieving hidden, meaningful, and unknown knowledge from a huge collection of data or database. In that, Frequent Itemset Mining 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.

Several approaches have been proposed for FIM, classified into two groups, they are CGAT (candidate generation and test) and other is without candidate generation that is FP-Growth [15]. 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. It has been extended and resulted with many approaches on FIM, including top-down approaches, Bottom-up approaches and combination of both is Pincer search [18,19]. However, their output is too big when the minimum support is low with redundancy. To avoid redundancy, maximal itemset concept is introduced. Maximal itemsets are the itemsets, whose frequency is sufficient and do not have any superset. Finding such itemsets usually requires the output of FIM. And then a simple comparison technique is required to compare the itemsets of FIM, is considered as a Naïve approach.

For example, consider Table 1, recorded with a list of 8 transactions, consider the minimum support min-sup=50% (count = 4). FIM { a:6, b:6, c:5, ab:5, bc:4, ac:4}. It can be seen that {a, ab, ac} are frequent and redundant, and {b, bc, c, ac} are also redundant.

Table 1: Illustration of Transactional Database

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

Several approaches have been carried out for deriving maximal frequent itemsets on the basis of the above two approaches. They are Depth project [2], MaxEclat[], Pincer search [18,19], MaxClique [24], MaxMiner [5], Mafia [8], GenMax [13] and FPMax [13]. 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.

We propose a Novel approach called Partition Based Single Scan Approach (PSS-MIM) for mining Maximal Itemsets, which solves the issues of FIM, other Maximal Itemset Mining algorithms. In PS-MIM, initially, all the transactions are divided into equal partitions, the possible itemsets are generated for each transaction. The possible itemsets are considered as Maximal and stored into hash table, if it is already not hashed. If they are already indexed, their counter will be incremented by one. After half of the partitions are visited, before storing into the hash, the candidates are tested whether they are maximal, frequent but not maximal or going to be infrequent. As a

Revised Manuscript Received on June 5, 2019

U. Mohan Srinivas, Research Scholar, Dept. of CSE, Acharya Nagarjuna University, Guntur, India.

Dr. E. Srinivasa Reddy, Professor, Dept. of CSE, Acharya Nagarjuna University, Guntur, India.

last step, the cumulated support of each itemset is considered to check the each of it is maximal or not. The purpose of this paper is to avoid post pruning to determine maximal itemsets without maintaining unnecessary candidate itemsets.

The remaining paper is structured as, review FIM and Maximal FIM algorithms in next section. Followed by the proposed novel approach and its description. Theoritical, Result Analysis are in continous sections, and concluded with the Conclusion section.

II. RELATED WORK

FIM is one of the fundamental data mining task among Mining Correlation among the items of database, Association Rules [3, 10, 11, 12 and 25], Classification [20], approximation [14], FIM in uncertain data [17,21]. Extension to the FIM is investigated and the resultant algorithms are Parallel [22], single scan algorithms [23]. The investigation on FIM has been carried out and resulted with many algorithms, such as Maximal, Minimal, Closed [9], Generators, and Sequential Patterns [1]. Mining Maximal itemsets become popular since it doesn't exhibit redundancy. Literature finds few work to find Maximal Itemsets, such as are MaxEclat [26], MaxClique [24], Pincer search [18,19], Maxminer [5], Depth project [2], Mafia [8], GenMax and FPMax [13].

Bayardo et al. introduced apriori kind of approach MaxMiner [5] algorithm to derive maximal itemsets. It is also not exception to huge memory. To enhance the performance, uses pruning strategy that is subset infrequency and closure checking that is superset frequency check to limit the usage of search space. Though it reduces search time and search space, it requires many passes to derive all the maximal itemsets. To aim at the issues of MaxMiner[5], Zaki et al. [26] has proposed MaxEclat and Max Clique. MaxEclat is based on equivalence concept, where each itemset maintains the occurrence list of transactions. In addition to that it maintains look ahead to identify Maximal itemsets. Max Clique algorithm is popular in graph applications that uses dynamic programming algorithm to find maximal cliques. Agarwal et al., [2] proposed a bitmap representation approach, which uses lattice and traverse with the combination of depth-first and Breadth-first traversal. In this approach, both superset and subset infrequency checking pruning strategies are used to avoid unnecessary computation and space. Burdick et al. [8] proposed an extension to the DepthProject that is MAFIA to mine maximal itemsets. It uses bitmap representation for itemsets, uses depth-first algorithm, subset, superset and Parent Equivalence pruning techniques.

Later, to reduce the multiple scans, Lin and Kedem [18,19] proposed Pincer search algorithm. It uses the combinations of bottom-up and top-down approaches to derive important knowledge, and also maintains infrequent items. It uses apriori and algorithm grows by using Maximum-Frequent-Candidate-Set (MFCS) at one end that is top, and Maximum Frequent Set (MFS) at other end bottom-up. It uses infrequent itemsets to ignore unnecessary itemsets. These kind of algorithms usually takes more scans when the data base is sparse.

The above approaches efficiently derive all the maximal itemsets. However, many passes required for Apriori based approaches and huge storage space is required for tree-based approaches. Hence very next section introduces a novel approach that derives all the maximal itemsets with a single pass.

III. PARTITION BASED SINGLE SCAN APPROACH FOR MAXIMAL ITEMSET MINING (PSS-MIM)

This section presents and discusses the proposed algorithm PSS-MIM, which is an extension of SS-FIM. It is continued with the theoretical analysis and result analysis in comparison with Apriori[], Maxminer [5] and MaxEclat [26].

Maximal Itemsets: Maximal itemsets are the frequent itemsets that do not have any supersets. These kind of itemsets are also called as the largest frequent itemsets. It can be denoted as MI ⊆ L, where MI is Maximal itemsets and L is frequent itemsets. From the definition, all maximal itemsets are also frequent. Hence, it can say that MI reduces the search space.

A.Description

PSS-MIM aims at reducing multiple scans over the database for deriving Maximal itemsets. The basic plot of the PSS-MIM is to partition into M equal partitions where the size of each partition is same as half of the minimum support, and generate all the candidates for each transaction. It assumes that the generated itemsets are maximal. 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.

The fundamental theme of contribution is to avoid the itemsets which are not going to become maximal. 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. Hence the computation for such kind of unnecessary itemsets is improved without losing information. In addition to that, it also checks superset checking to decide maximal itemsets, itemsets are maintained in a list which are frequent but not maximal. Such itemsets are removed from hash. Finally, it gives only maximal itemsets in hash table.

PSS-MIM is complete, because all maximal frequent itemsets are derived directly from the candidate itemsets which are generated directly from the given TDB, whose support is ≥minsup. After visiting minsup of TDB, the itemsets that are not in hash table, then there is no chance of getting the frequency of minsup. Hence it is complete. Algorithm 1 describes the step by step activities of PSS-MIM.

Procedure PSS-MIM for mining maximal itemsets is presented below. As a first step, It takes transactional database TDB as input, and minsup given by the user and it divides TDB into M partitions which are equally size shown as first statement. Size of each partition is the half of minsup of TDB. In second step, it generates all the possible candidate maximal itemsets for each transaction. In step 3, it uses data structure hash table for candidate itemsets with their support value w.r.t the

conditions mentioned in algorithm. 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. At the end, it returns all the itemsets whose support value is ≥ minsup.

Procedure: PSS-MIM

Input: TDB, min-sup

Output: LM: List of Maximal 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 Mi, for each transaction, generate all the possible candidate itemsets to C.

Step 3: for each c of C, consider as maximal itemset and do the following steps

(i) If c is found to be a subset of any frequent itemset of hash i.e c⊂hash(c), then ignore c and delete hash index if it is already indexed.

(ii) If c doesn't have any index in hash, create index and assign with 1 as a support.

(iii) If c is already created, then cumulate its support by 1.

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 > (|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 an itemset *c* is infrequent, then the itemsets that are supersets of *c* are also infrequent. As per the Anti-monotonic property described in [2, 3], the above property states true.

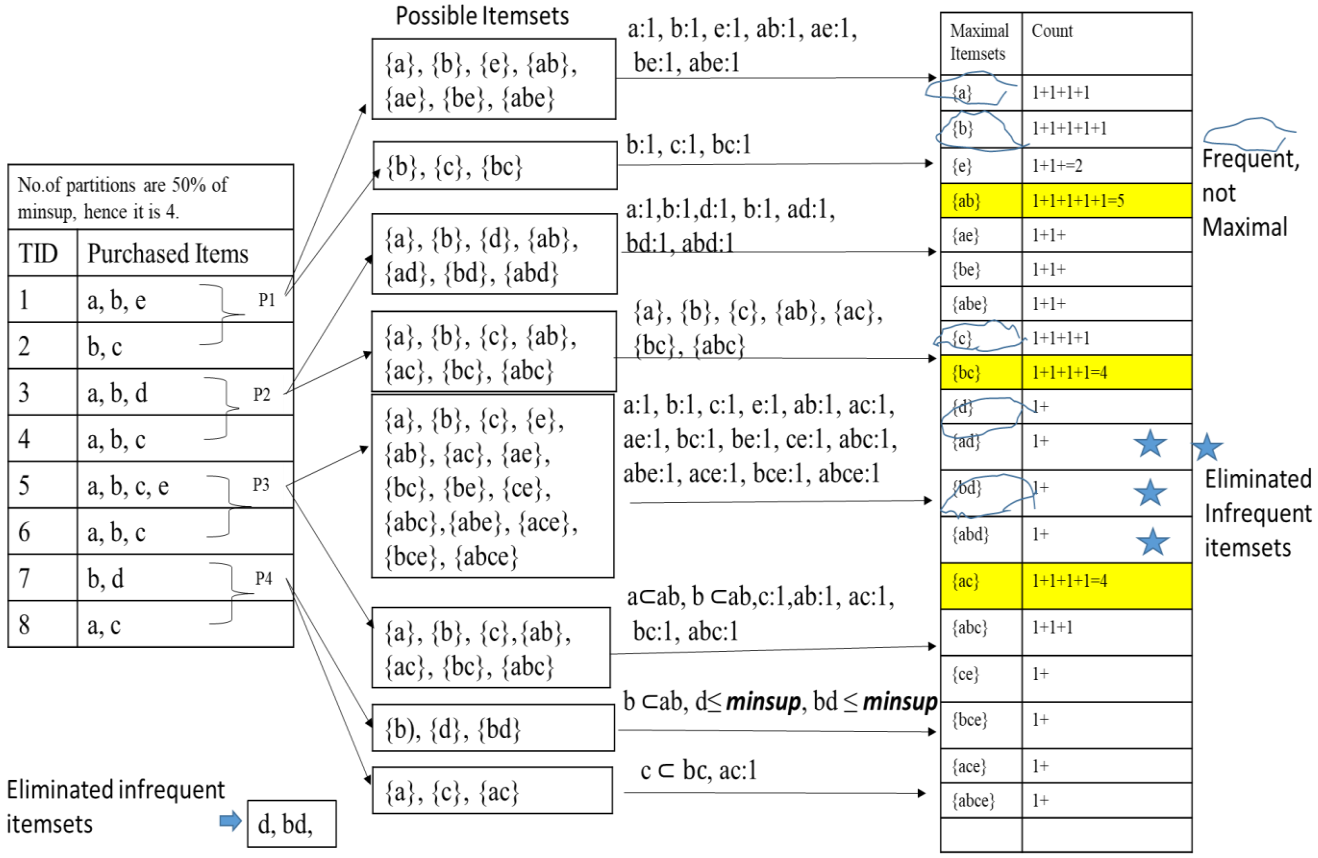

Figure 1: PSS-MIM illustration of Table1

B. Illustration

Figure 1 shows the pictorial representation of PSS-FIM algorithm execution for the 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 contains {T1, T2}, P2 {T3, T4}, P3 {T5, T6}, and P4 {T7, T8}.

Step 2: the possible maximal itemsets from transaction T1 {a, b, e} are {a}, {b}, {e}, {ab}, {ae}, {be}, and {abe}. Initially, the hash table h is empty, then all these itemsets are stored into the h and support is initialized to 1. Same procedure is repeated until half of the partitions are visited. For T6 second of P3 is {a,b,c}, and the possible itemsets are { {a}, {b}, {c}, {bc},{ac},{ab}, {abc}}. It is observed that itemset {a} □ {ab:4}, and as per the definition of Maximal, itemset {a} will not become maximal. Hence such kind of itemsets ({a}, {b} □ {ab:4}) are removed from hash table and maintained in a list FM. And the rest of the itemsets are inserted into hash h. Transaction T7 {b,d}, possible itemsets are { {b}□

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

{ab:4}, {d}, {bd}}, itemset Sup({d},h) is 1 and the remaining transactions are 1, hence Sup({d})<minsup. As per the Heuristic 2, such kind of itemsets will be infrequent and are not maintained in hash. And also the supersets of {d} in hash are deleted from hash w.r.t heuristic 3. After processing all transactions, the final result contains {ab}, {bc} and {ac} are the maximal itemsets.

C.Theoretical Analysis

The time complexity of PSS-MIM is determined from (i) possible itemsets generation cost and (ii) Maximal frequent itemsets identification cost. The cost of candidate generation of each transaction Ti is $2^{|Ti|}$ -1, and for TDB is $\sum_{i=1}^{N} 2^{T_i} - 1$, where N is |TDB|. Partition heuristic of PSS-MIM discards some itemsets which are frequent and not maximal, denoted as IFM. And IX are the itemsets which are not frequent. Hence it is $\sum_{i=1}^{N}(2^{T_i} - 1 - I_{FM} - I_{Ix})$. the complexity of possible itemset generation is $O(N(2^p - 1 - I_X - I_{FM}))$, -- (1)

Where *p* is the possible number of itemsets for each transaction. The second one, frequent itemsets determining cost, the hash table h looks at each index to compute support, then operation is $N(2^p - 1 - I_X - I_{FM})$

The total running cost of PSS-MIM is $O(2N(2^p - 1 - I_X - I_{FM}))$, -- (2)

Whereas the complexity of Apriori is O ($N \times M^2$), for maximal itemsets is $O(N^2)$. Hence complexity is $O((N \times M^2)+(N^2))$. -- (3)

D.Result Analysis

To analyze the performance of PSS-MIM, experiments are conducted on the data sets used in [16]. Bolt dataset contains 2178 transactions, where each transaction with the range of items from 8 to 16.

Medium sized databases BMS_Webview and retail whose size varies from 59000-100000, where the possible items range between [500, 16000]. The other type of dataset is a large database BMS-POS, which contains 500000 transactions. Each transaction is recorded with an average 2.5 items out of 1660 items. The investigation on the above datasets are carried on computer with 4GB Ram, processor Intel I3.

The running time comparison of PSS-MIM, Apriori-maximal which is an extension of Apriori [2], Maxminer [5] and MaxEclat [26] are presented in Table 2. Since the proposed method does mining maximal itemsets with a single scan, the performance of Apriori is equally same as other approaches. However for medium and large database, PSS-MIM give better performance that is more than twice compared to other approaches. Hence, PSS-MIM is efficient when the database is dense.

Table 1 Run time comparison of PSS-MIM in seconds

Data set	Apriori-Maximal		MaxMiner		MaxExclat		PSS-MIM	
	Time (s)	Scans	Time (s)	Scans	Time (s)	Scans	Time (s)	Scans
Bolts	6	2	5	2	5	2	4	1
Sleep	9	2	8	2	8	2	8	1
Pollution	25	2	20	2	18	2	15	1
Basket ball	20	2	18	2	17	2	15	1
Quake	34	2	30	2	30	2	25	1
BMS-Web View-1	1200	2	500	2	400	2	200	1
BMS-Web View-2	5120	2	3000	2	2500	2	1500	1
Retail	5420	2	4000	2	3500	2	2100	1
Connect	3400	2	2500	2	2300	2	1400	1
BMS POS	11450	2	6000	2	5500	2	3150	1

Figure 2 is recorded with observations of the PSS-MIM with respect to the various threshold values on small dataset Bolt. It is observed that the growth of the runtime of PSS-MIM is constant compared to other approaches which takes two scans over the database.

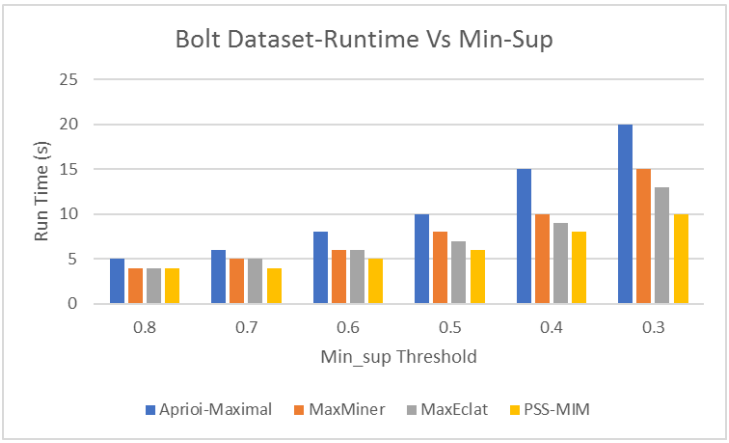

Figure 2: Runtime comparison of PSS-MIM w.r.t various threshold values over Bolt Database

Figure 3 is recorded with runtime observations of the PSS-MIM with respect to the various threshold values on recognized as Average Sized dataset BMS-Web View-1. It is observed that the proposed approach PSS-MIM takes less time compared to others, and the growth of the runtime of PSS-MIM is constant compared to other approaches which takes two scans over the database.

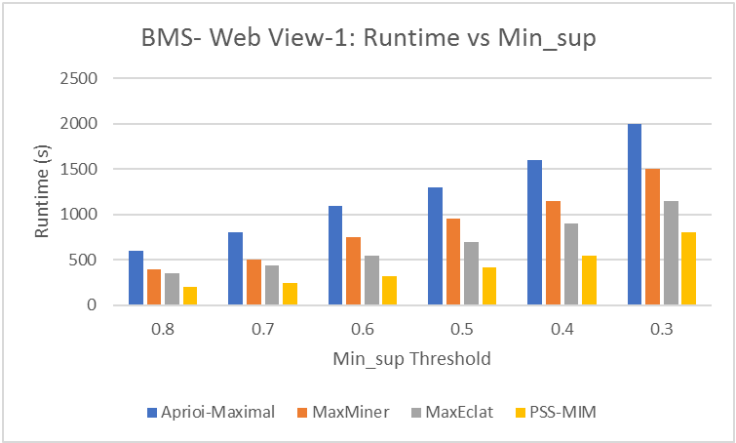

Figure 3: Runtime comparison of PSS-MIM w.r.t various threshold values over BMS-Web View-1 Database

Figure 4 is recorded with runtime observations of the PSS-MIM with respect to the various threshold values on recognized as large dataset Retail. It is observed that the growth of the runtime of PSS-MIM is constant compared to other approaches which takes two scans over the database.

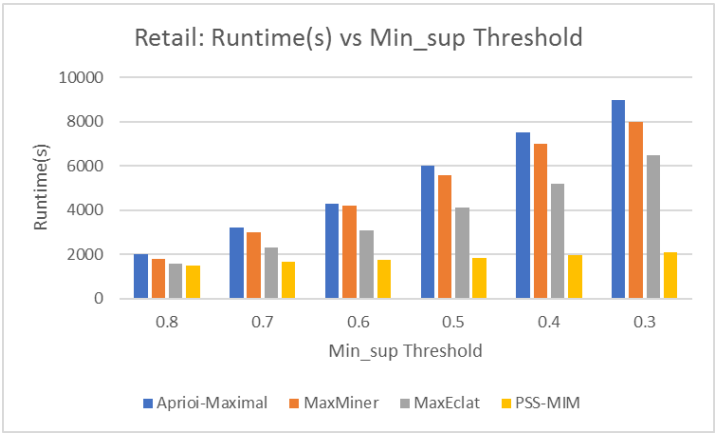

Figure 4: Runtime comparison of PSS-MIM w.r.t various threshold values over Retail Database

IV. CONCLUSION

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

In further, it is evident that more heuristics can be imposed to reduce the running time and 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. Amphawan, K., Lenca, P., Surarerks, A.: Efficient mining top-k regular-frequent itemset using compressed tidsets. In: Cao, L., Huang, J.Z., Bailey, J., Koh, Y.S., Luo, J. (eds.) PAKDD 2011. LNCS (LNAI), vol. 7104, pp. 124–135. Springer, Heidelberg (2012). doi:10.1007/978-3-642-28320-8 11.
5. Bayardo, R.J. (1998) 'Efficiently mining long patterns from databases' in Proceedings of ACM SIGMOD Conference on Management of Data, pp. 85–93, New york, USA.
6. Borgelt, C.: Frequent itemset mining. Wiley Interdisc. Rev.: Data Min. Knowl. Discov. 2(6), 437–456 (2012).
7. Brin, S., Motwani, R., Ullman, J.D., Tsur, S.: Dynamic itemset counting and implication rules for market basket data. In: ACM SIGMOD Record, vol. 26, no. 2, pp. 255–264. ACM, June 1997.
8. 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.
9. 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).
10. 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).
11. 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).
12. 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).
13. Grahne, G., Zhu, J.: Fast algorithms for frequent itemset mining using FP-trees. IEEE Trans. Knowl. Data Eng. 17(10), 1347–1362 (2005).
14. Guvenir, H.A., Uysal, I.: Bilkent university function approximation repository (2000). http://funapp.CS.bilkent.edu.tr/DataSets. Accessed 12 Mar 2012.
15. 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
16. Hegland, M.: The apriori algorithm tutorial. Math. Comput. imaging Sci. Inf. Process. 11, 209–262 (2005).
17. Leung, C.K.-S., Mateo, M.A.F., Brajczuk, D.A.: A tree-based approach for frequent pattern mining from uncertain data. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS (LNAI), vol. 5012, pp. 653–661. Springer, Heidelberg (2008). doi:10.1007/978-3-540-68125-0 61.
18. Lin, D. and Kedem, Z.M. (1998) 'Pincer-Search: A New Algorithm for Discovering the MaximumFrequent Set', in Proceedings of the 6th International Conference on Extending Database Technology (EDBT), Valencia .pp. 105-119.
19. 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.
20. Liu, B., Hsu, W. and Ma, Y. (1998) 'Integrating Classification and Association rule mining', in Proceedings of the 9th ACM SIGKDD international conference on Knowledge Discovery and Data mining, pp 80-86.
21. Luna, J.M., Pechenizkiy, M., Ventura, S.: Mining exceptional relationships with grammar-guided genetic programming. Knowl. Inf. Syst. 47(3), 571–594 (2016).
22. Mueller, A.: Fast sequential and parallel algorithms for association rule mining: a comparison. Technical report CS-TR-3515, University of Maryland, College Park, August 1995.
23. 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.
24. Zaki, M.J. (2000) 'Scalable algorithms for Association Rule Mining', in IEEE transactions on Knowledge Discovery and Data Engineering, Vol 12, No. 3, pp. 372-390.
25. 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).
26. Zaki, M. J., Parthasarathy, S., Ogihara, M., Li, W., Stolorz, P., & Musick, R. (1997). Parallel Algorithms for Discovery of Association Rules. Scalable High Performance Computing for Knowledge Discovery and Data Mining, 5–35. doi:10.1007/978-1-4615-5669-5_1.

AUTHORS PROFILE

U. Mohan Srinivas received B.Tech in Electronics & Communication Engineering from Nagarjuna University, India in 1991, M.Tech. in Computer Science and Engineering from Jawaharlal Nehru Technological University, Kakinada, India in 2004. He is currently a Ph.D. **Research Scholar** in Computer Science and Engineering at Acharya Nagarjuna University under the guidance of Prof. E. Sreenivasa Reddy. He is a professional member of ACM and CSI. He is currently active in the fields of Data Mining, Signal/Image Processing, Artificial Intelligence and Pattern recognition.

Dr. E. Sreenivasa Reddy graduated in B.Tech (ECE) from Nagarjuna University, India in 1988, M.S. degree from Birla Institute of Technology and Scince, India in 1997, M.Tech (CS) from Visveswaraiah Technological University, India in 2000 and Ph.D in Computer science from Acharya Nagarjuna Univeristy, India in 2008. Currently he is guiding many Ph.D scholars for several universities. He has published many papers in National, International Journals and conferences. He is the senior member of IEEE. He is currently active in the research fields of Image Processing, Biometrics and Pattern recognition.