Latest Survey On Frequent Pattern Mining: Mine The Frequent Patterns From Transaction Database

SURATI Sandip B.
Vivekanand College for BCA
Near Saroli Jakatnaka,
Jahangirpura, Surat
suratisandip@yahoo.co.in

DESAI Apurva A.
Department of Computer Science
Veer Narmad South Gujarat
University, Surat
aadesai@vnsgu.ac.in

Abstract

Frequent patterns mining is one of the most important concepts in data mining. In last decades, lots of research has been done in area of frequent pattern mining. Frequent patterns are used in many data mining task such as association rules, correlations, clusters etc… This paper surveys latest frequent pattern mining algorithms and compare them to know their disadvantages and advantages over others and to understand various problems still to be solved.

Keywords

Research on frequent pattern mining, frequent itemset mining, Web Usage Mining, Mining frequent patterns

1. Introduction

Data Mining is most important research area in 21st centuary, In recent years the database size is increase rapidly. Web Usage Mining is used to find the usage pattern from web data, in order to understand and better serve the needs of web based applications [1]. Association rule is used to find the frequent patterns from transaction database and generate association rules from the same. The motivation to search association rules came from the need to analyze supermarket transaction data [2]. Association rule mining is divided in two steps. 1) Find the frequent patterns and 2) generate association rules. The second step of Association rule mining is easy. The first step of association rule mining is challenging task. Lots of research has been done in area of frequent pattern mining. Various algorithms have been proposed to generate frequent itemsets[3]. Researchers have proposed various improved algorithms for generating frequent itemsets. These algorithms differ in their ways by how they traverse the database, how they handle the database, it means how many passes they make on database. This paper is an overview of various algorithms proposed by researchers.

2. Literature Survey

2.1 AIS Algorithm

The first algorithm was AIS Algorithm [1]. It is multi-pass algorithm in which candidate itemsets are generated during scanning the database by extending known-frequent itemsets with items from each transaction. There are mainly two disadvantages of the AIS algorithm. 1) It generates large number of candidates that later turn out to be infrequent. 2) The data structures were not specified.

2.2 SETM Algorithm

The SETM algorithm, which uses SQL to generate frequent itemsets. The SETM algorithm represents each member of the candidate/frequent itemsets in the form <TID, itemset>. In which TID is the unique identifier of a transaction. The disadvantage of SETM algorithm is that it makes multiple passes over the database. The main problem in SETM is the number of candidate itemsets. With each candidate itemset the TID is associated, so it takes more space to store a large number of TIDs [4].

2.3 Apriori Algorithm

Apriori is a classical algorithm for learning association rule. It performs better than AIS and SETM algorithm. Apriori completely incorporate the subset frequency based pruning optimization it means, it does not process any itemset whose subset is known to be infrequent. It uses data structure called hashtree to store the count of candidate itemsets. The main disadvantage of the Apriori is the complex candidate generation process that uses most of the time, space and memory. It generates huge number of candidate set. Another disadvantage is multiple scan of the database.

There are many improvements of Apriori. Some improvements are partitioning technique [5], sampling approach [6], dynamic itemset counting [7], Novel method for counting the occurrences of itemsets [8], CARMA [9], hashing technique [10], incremental mining [11], Efficient parallel mining for association rules [12], Mining sequential patterns [13], Parallel algorithm for discovery of association rules [14], an integrating mining with relational database systems [15] A tight upper bound on number of candidate patterns[16], Record Filter Algorithm [17], Intersection Algorithm [17], Proposed Algorithm based on Apriori [17], DFPMT Algorithm [18], Improved Frequent Pattern Algorithm [19], DHP Algorithm [20]. Following are some important variations of Apriori algorithm in brief.

2.4 Sampling Algorithm

The sampling algorithm first mines a random sample of the database to get itemsets that are frequent within the sample. These itemsets could be considered as a representative of the actual frequent itemsets in applications where approximate mining results are sufficient [6]. To get the accurate mining results, this approach needs one or two scans over the entire database. This algorithm follows a tuple-by-tuple approach, therefore like Apriori, it suffers from drawback as of Apriori.

2.5 DIC Algorithm

The DIC algorithm is also known as non-level-wise algorithm [7]. In this algorithm, the candidates are generated and removed after every M transaction; here M is a parameter to the algorithm. It is a multi-pass algorithm. The DIC algorithm is also suffers from the drawbacks of tuple-by-tuple approaches.

2.6 CARMA Algorithm

In CARMA (Continuous Association Rule Mining Algorithm) algorithm, the computation of frequent itemsets is online. It displays the current association rules to the user and permits the users to change the parameters and also permit to change minimum confidence and minimum support, at any transaction during the first scan of the database [9]. The algorithm is two-pass algorithm. It offers features for dynamically generating and removing candidates after each record of the database is processed. The algorithm do not perform consistently better than Apriori, its memory utilization was less by an order of magnitude.

2.7 Record Filter Algorithm

Record Filter algorithm count the support of candidate set in the transaction record whose length is greater than or equal to the length of candidate set, because candidate set of length k , cannot exist in the transaction record of length k-1 , it may exist only in the transaction of length greater than or equal to k. Record filter approach proved better than classical Apriori Algorithm [17].

2.8 Intersection Algorithm

In Intersection approach, to calculate the support we count the common transaction that contains in each element's of candidate set, by using the intersect query of SQL. This approach requires very less time as compared to classical Apriori. Intersection approach proved better than Record filter approach [17].

2.9 Proposed Algorithm based on Apriori

This algorithm uses the concept of both algorithm i.e. Record filter approach and Intersection approach in Apriori algorithm. The algorithm uses the set theory concept of intersection with the record filter approach. In proposed algorithm, to calculate the support, it count the common transaction that contains in each element's of candidate set, with the help of the intersect query of SQL. The algorithm applied a constraints to consider only those transaction that contain at least k items, not less than k in process of support counting for candidate set of k length. This approach requires very less time as compared to all other approaches. Proposed algorithm proved that it is much better than other frequent pattern mining algorithm i.e Apriori, Record Filter and Intersection algorithm [17].

2.10 DFPMT Algorithm

DFPMT is stands for Dynamic Approach for Frequent Patterns Mining using Transposition of database for mining frequent patterns which is based on Apriori algorithm and used Dynamic function for Longest Common Subsequence. In DFPMT, the database stores in transposed form and in each iteration database is filter /reduce by generating LCS of transaction id for each pattern [18].

2.11 Improved Frequent Pattern Algorithm

Improved frequent pattern algorithm mine the frequent patterns in large datasets using transposition of the database with minor modification of the Apriori-like algorithm. The main advantage is database stores in transposed form and in each iteration database is filtered and reduced by generating the transaction id for each pattern. The algorithm reduces the huge computing time and also decreases the database size [19].

2.12 DHP Algorithm

The DHP algorithm is also known as the Direct Hashing and Pruning method [20]. It proposes two main optimizations technique to speed up the algorithm. First optimization is to prune the candidate itemsets in each iteration. The second optimization is to trim the transactions to make the support-counting process more efficient.

2.13 FP-Tree Based Algorithm

The FP-growth method mines the complete set of frequent itemsets without candidate generation [21]. This method is based on the divide and-conquer principle. The database is compressed into a frequent pattern tree, also called as FP-tree, which retains the itemset association information. We can find the frequent patterns by mining the FP-tree. The disadvantage of FP-tree is that the construction of the FP-tree is a very time consuming process. The second disadvantage is it does not offer flexibility and reusability of computation during mining process.

There are many alternatives and improvement to the FP-growth approach, including depth-first generation of frequent itemsets [22], Mining frequent itemsets with convertible constraints [23], Mining frequent item sets by opportunistic projection [24], On computing, storing and querying frequent patterns [25], an array-based implementation of prefix-tree-structure for efficient pattern growth mining [26], CATS Algorithm [27], AFPIM Algorithm [28], CAN Tree[29], CP-Tree [30], Efficient Prefix Tree [31]. Following are some improvement of FP-growth algorithm in brief.

2.14 CATS Algorithm

The CATS algorithm mine frequent patterns in an incremental manner [27]. The proposed tree structure is an improvement on FP-tree. It extracts frequent patterns without candidate sets generation. In this algorithm, the first transaction in database is added to the tree’s root. For subsequent transactions, the items within the transaction are compared with the items in the tree to identify shared items. If there is any item in common between tree nodes and the transaction, the transaction is merged with the node that has the highest frequency level. Then, the remainder of the transaction is added to the merged nodes. This process is recursively repeated until all common items are discovered.

2.15 AFPIM Algorithm

In AFPIM algorithm, PreMinsup is considered whose values are set less than the Minsup. Since, items are ordered based on the number of events, the insertion, deletion or modification of transactions may affect the frequency and order of the items. Items in the tree are adjusted when the order of the items changes. The AFPIM algorithm swaps such items by applying bubble sort method that involves huge calculation [28].

2.16 CanTree Algorithm

Can-tree algorithm needs only one database scan. In Can-Tree algorithm, items are ordered on the basis of a canonical standard (for e.g. alphabetical) which can be determined by the user. Therefore, any changes in frequency, which is caused by incremental updates (like insert, delete, or update transactions), will not affect the order of items in the Cantree. Therefore, new transactions are inserted into the Cantree without swapping any tree nodes [29].

2.17 CP-Tree Algorithm

In CP-Tree Algorithm, all the transactions are inserted into the tree as per predefined item order. This predefined item order is maintained in a list, called I-list. After inserting some of the transactions, if the item order of the I-list differs from the current frequency-descending item order to a predefined degree, the CP-tree is restructured by method called the branch sorting. Then, the item order is updated with the current list [30].

2.18 Efficient Prefix Tree

CP-Tree has one problem. During construction of CP-tree items are sorted in descending order of previous insertion phase, then its restructuring can be very costly. To solve this problem, efficient prefix tree structure is used to reduce the time of restructuring. The tree is created based on the frequency of last items and it requires just one database scan.

2.19 MAXCLIQUE Algorithm

The above discussed algorithms mine the database which is in horizontal data layout. The other way of mining, in which data presented in vertical data format (i.e., {item: TID_set}). Each item is shown with list of TIDs (Transaction Ids), in which item appears. The MaxClique algorithm is designed to efficiently mine databases which are in a vertical layout [14].

2.20 Equivalence Class Transformation (ECLAT)

The ECLAT algorithm mines the database which is in the vertical data format [32]. In the first scan of the database, it generates the TID_set for each single item. The frequent (k+1)-itemset has grown from a previous k-itemset. The computation is done by intersection of the TID_sets of the frequent k-itemsets to compute the TID_sets of the corresponding (k+1)-itemsets. This process repeats, until no frequent itemsets or no candidate itemsets can be found. The disadvantage of this algorithm is huge number of candidate generation.

2.21 Viper

The above vertical mining algorithms have various restrictions regarding database shape, size, content or the mining process. The viper algorithm does not have any such restrictions. It includes many optimizations to enable efficient processing. The viper algorithm was outperforming to earlier vertical mining algorithms [33].

2.22 Pattern Mining Algorithm (PM)

Pattern mining algorithm generates frequent patterns using simple processing technique. It works with only one database scan. Pattern mining algorithm works in two phases. (1) Generate the subset list. (2) Generate the frequent item sets using the subset list. The disadvantage is that it takes more time to generate the frequent patterns [34].

3. Conclusion

In last decades, number of researchers developed several algorithms, compared them and tried to solve the frequent itemset problem as efficiently as possible, like minimum number of database scan, low memory usage, less run time, generate frequent patterns without candidate generation etc.. Each algorithm has its own advantages and limitations. Some algorithms work on horizontal data layout and others on vertical data layout. This paper has discussed most popular apriori algorithm, fp-growth algorithm and also many improved algorithms based on apriori and fp-growth.

This survey highlighted many algorithms which made a significant contribution to improve the efficiency of frequent pattern mining.

Acknowledgement

We acknowledge UGC for Special Assistance Program (SAP) for Natural Language Processing and Data Mining (file number is F.3-48/2011) under which this work has been done.

References

[1] J. Shrivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web Usage mining: Discovery and applications of Usage patterns from web data. SIGKDD Explorations, 1(2), 2000.

[2] R. Agrawal, T. Imielienski, and A. Swami. 1993. Mining Association Rules between Sets of Items in Large Databases. Proc. Conf. on Management of Data, 207–216. ACM Press, New York, NY, USA 1993.

[3] S. Kotsiantis, D. Kanellopoulos. Association Rules Mining: A Recent Overview, GESTS International Transactions on Computer Science and Engineering,Vol.32 (1), 2006, pp. 71-82.

[4] R. Agrawal and R. Srikant. 1994. Fast Algorithms for Mining Association Rules. Proc. 20th Int. Conf. on very Large Databases (VLDB 1994, Santiago de Chile), 487–499. Morgan Kaufmann, San Mateo, CA, USA 1994.

[5] A. Savasere, E. Omieccinski and S. Navathe, 1995. An efficient algorithm for mining association rules in large databases. Proceedings of the 21st International Conference on Very Large Databases, September 11-15, 1995, Zurich, Switzerland, pp: 432-443.

[6] H. Toivonen, 1996. Sampling large databases for association rules. Proceedings of 22th International Conference on Very Large Databases, September 3-6, 1996, Bombay, India, pp: 134-145.

[7] S. Brin, R. Motwani, J.D. Ullman and S. Tsur, 1997. Dynamic itemset counting and implication rules for market basket data. Proc. 1997 ACM SIGMOID Int. Conf. Manage. Data, 26: 255-264.

[8] A. Tiwari, R.K. Gupta and D.P. Agrawal, 2009. A novel algorithm for mining frequent itemsets from large database. Int. J. Inform. Technol. Knowl. Manage., 2: 223-229.

[9] C. Hidber, 1999. Online association rule mining. ACM SIGMOD Rec., 28: 145-156.

[10] J.S. Park, M.S. Chen and P.S. Yu, 1995. An effective hash-based algorithm for mining association rules. ACM SIGMOD Rec., 24: 175-186.

[11] D.W. Cheung, J. Han, V.T. Ng and C.Y. Wong, 1996. Maintenance of discovered association rules in large databases: An incremental updating technique. Proceedings of International Conference on Data Engineering, February 26-March 1, 1996, New Orleans, Louisiana, pp: 106-114.

[12] J.S. Park, M.S. Chen and P.S. Yu, 1995. Efficient parallel mining for association rules. Proceedings of the 4th International Conference on Information and Knowledge Management, Nov. 29-Dec. 2, Baltimore, MD., pp: 31-36.

[13] R. Agrawal and R. Srikant, 1995. Mining sequential patterns. Proceedings of the 11th International Conference on Data Engineering, March 6-10, 1995, Taipei, Taiwan, pp: 3-14.

[14] M.J. Zaki, S. Parthasarathy, M. Ogihara and W. Li, 1997. Parallel algorithm for discovery of association rules. Data Min. Knowl. Discov., 1: 343-374.

[15] S. Sarawagi, S. Thomas and R. Agrawal, 1998. Integrating association rule mining with relational database systems: Alternatives and implications. ACM SIGMOD Rec.,27:343-354.

[16] F. Geerts, B. Goethals and J. Bussche, 2001.A tight upper bound on the number of candidate patterns. Proceedings of the 2001 International Conference on Data Mining, Nov. 29-Dec. 2, San Jose, CA., pp: 155-162.

[17] D. N. Goswami, Anshu Chaturvedi, C. S. Raghuvanshi, 2010. An algorithm for frequent pattern mining based on Apriori. International Journal on Computer Science and Engineering Vol. 02, No. 04, 2010, 942-947.

[18] S. Joshi. An Implementation of Frequent Pattern Mining Algorithm using Dynamic Function. International Journal of Computer Applications (0975 – 8887) Volume 9-No.9, November 2010.

[19] D. Gunaseelan, P. Uma. An Improved Frequent Pattern Algorithm for Mining Association Rules. International Journal of Information and Communication Technology Research, Volume 2 No. 5, May 2012, ISSN 2223-4985.

[20] J. S. Park, M. S. Chen, and P. S. Yu. An effective hash based algorithm for mining association rules. In Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data, volume 24(2) of SIGMOD Record, pages 175–186. ACM Press, 1995.
[21] J. Han, J. Pei, Y. Yin and R. Mao, 2004. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Mining Knowledge Discovery, 8: 53-87.
[22] R.C. Agarwal, C. Aggarwal and V.V.V. Prasad, 2001. A tree projection algorithm for generation of frequent item sets. J. Parallel Distributed Comput., 61: 350-371.
[23] J. Pei, J. Han and L.V.S. Lakshmanan, 2001. Mining frequent itemsets with convertible constraints. Proceedings of the 17th International Conference on Data Engineering, April 2-6, Heidelberg, Germany, pp: 433-332.
[24] J. Liu, Y. Pan, K. Wang and J. Han, 2002. Mining frequent item sets by opportunistic projection. Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery in Databases, July 23-26, Edmonton, Canada, pp: 239-248.
[25] G. Liu, H. Lu, W. Lou and J.X. Yu, 2003. On computing, storing and querying frequent patterns. Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Aug. 24-27, Washington, DC., pp: 607-612.
[26] G. Grahne and J. Zhu, 2003. Efficiently using prefix-trees in mining frequent itemsets. Proceedings of the 2003 ICDM International Workshop on Frequent Itemset Mining Implementations, Melbourne, FL., pp: 123-132.
[27] Cheung, W. and O.R. Zaiane. Incremental mining of frequent patterns without candidate generation or support constraint in Seventh IEEE international database engineering and applications symposium (IDEAS'03). 2003.
[28] Koh, J.L. and S.F. Shieh, An efficient approach for maintaining association rules based on adjusting FP-tree structures1. Lecture Notes in Computer Science, 2004: p. 417-424.
[29] Leung, C.K.S., et al., CanTree: a canonical-order tree for incremental frequent-pattern mining. Knowledge and Information Systems, 2007. 11(3): p. 287-311.
[30] Tanbeer, S.K., et al., Efficient single-pass frequent pattern mining using a prefix-tree. Information Sciences, 2009. 179(5): p. 559-583.
[31] M. Hamedanian, M. Nadimi, M. Naderi. An Efficient Prefix Tree for Incremental Frequent Pattern Mining. International Journal of Information and Communication Technology Research, Volume 3 No. 2, February 2013, ISSN 2223-4985.
[32] M.J. Zaki, 2000. Scalable algorithms for association mining. IEEE Trans. Knowl. Data Eng., 12: 372-390.
[33] P. Shenoy, J. Haritsa, S. Sudarshan, G. Bhalotia, M. Bawa and D. Shah, 2000. Turbo-charging vertical mining of large databases. Proceedings of ACM SIGMOD International Conference on Management of Data, MAY 16-18, Dallas, TX., pp: 22-33.
[34] S. Surati, A. Desai. Pattern Mining (PM) Algorithm, VNSGU Journal of Science and Technology, Vol.3, No.2, March, 2012, 64-72, ISSN: 0975-5446.