ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

IMPLEMENTATION OF DIC USING ASSOCIATION RULE MINING ALGORITHM

Rasika Vyawhare, Priyanka Mandape, Sachin Chawhan
* Computer science and Engineering R.T.M.N.U Nagpur

ABSTRACT

A distributed algorithm is based on Dynamic Item- set Counting (DIC) using frequent itemset . Since DIC perform a Apriori-based algorithms in the number of passes of the database. Hence for reducing the total time taken to obtain the frequent data itemsets. The advantage of Dynamic Itemset Counting is that it will provide to starting from the counting of an itemset at the different site as soon as they become frequent at atleast one site. Hence, our algorithm shows remarkable improvement in the amount of time taken because of reduction in the number of passes of the database and comparatively lesser number of candidates are generated. Before Distributed frequent itemset count an association rule mining have basically used. This algorithms based on DIC and Apriori for large dataset and thus implementing the first algorithm which is based on DIC.

In the paper we have explain one of the useful and efficient algorithms of Association rule mining called as DIC algorithm. Association rule mining algorithm of data is used in all real life applications of business and industry. Using this application they provide effective results rather than traditional results. Association rules mining algorithm are the main technique and concepts for large data items. DIC algorithm is a classical algorithm in which Lots of algorithms are mined for large association rules. Therefore they (change/transformation) are proposed on basis of DIC algorithm. For using this traditional algorithms they are not sufficent. The main motive of this paper is to understand the concept of association rule mining algorithm and what methods are used for implement of DIC algorithm and thus improved DIC algorithm in future use.

For using these concept DIC are manage for large dataitemset.These algorithm has ruled for implementation of association rule mining algorithm.The main reason for using these DIC algorithm is that they maintain a constant speed at which data is used.For large application it is used for data maintaince and security protection.Hence DIC used in online shopping.It also need more capacity management for large dataset for frequent item.

KEYWORDS: Abstract of DIC algorithm,association rule mining,DIC agorithm Step of DIC,Example,How DIC works,Relatedwork,Advantages, Applications,Conclusion,References.

INTRODUCTION

DIC has been used extensively used for the classical problem of market basket analysis where it is required to find out the buying habits of a customers.For Determining what products customers are likely to buy together can be very useful for planning and marketing. Association rules mining algorithm are used to show the relationships between these data items count.

Most of the algorithms depend on the discovery of frequent itemsets for generation of association rules mining algorithm. Since the total number of itemsets are exponential in terms of the number of items, it is not possible to count the frequencies of these sets by visualising the database in just one pass.

Different algorithms are used for the discovery of association rules mining algorithm which aim that for reducing the number of passes by generating candidate sets, which are likely to be frequent itemsets. They attempt us to eliminate data in frequent item sets as early as possible.

Implementation of association rule mining algorithm is based on DIC in which data mining is a technique that helps to extract important data from a large database. This process of sorting large amounts of data and picking out relevant information through the use of DIC algorithm. As more data is needed, the large amount of data is doubling for every three years.Therefore,data mining is becoming an increasingly important tool for transforming data into

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

large information. Data mining techniques are the result of a long process of research and product development. This evolution began when business data was first stored on computers,and continued with the improvements in large data access, and more recently, developed technologies that allow users to navigate through their data in real time applications. Data mining takes this evolutionary process beyond retrospective data access and navigation to prospective and proactive information delivery on time management system.

ASSOCIATION RULE MINING

Let I=i1,i2,...iN be a set of items. Let D be a database of transactions, where each transaction T consists of a set of items such that T ⊆I. The support of an itemset X is the number of transactions which the itemset occurs as a subset. An itemset is frequent or large if its support is more than some user defined minimum support threshold δ. Thus support is the number of transactions in the database that contain the itemset X. An association rule is an implication of the form X ⇒ Y where X ⊂ I,Y ⊂ I and X∩Y = ∅. The rule X ⇒ Y holds in the transaction set D with confidence c if c% of transactions in D that contain X also contain Y. The rule X ⇒ Y has support s in the transaction set D if s% of transactions in D contain X ∪ Y. The problem of mining association rules is to generate all association rules that have a certain user- defined minimum support and confidence. Several centralized algorithms exist for Association Rule Mining. One of the first algorithms is Apri- ori, on which most of the parallel algorithms are based. Apriori is an iterative, level-wise algorithm which uses a bottom-up search starting with the counting of frequent 1- itemsets. It generates these itemsets after a complete scan of the database. It then uses a self-join to find the 2-itemsets from the frequent 1-itemsets. It then scans the database to find the frequent 2-itemsets and continues this process till the maximal itemsets are generated. The number of passes is equal to the size of the maximal n-itemset. It uses the large itemset property that is any subset of a large itemset must be large. The large itemsets are also said to be downward closed because if an itemset satisfies the minimum support requirements so will its subsets. Hence, if we know that an itemset is small then we need not generate its supersets as candi- dates because they will also be small.The performance of Apriori directly depends on the length of the longest frequent itemset. A remarkable breakthrough in sequential algorithms was achieved by the Dynamic Itemset Count- ing (DIC) algorithm which represents a shift in the method in which frequent itemsets are generated. Since Dynamic Itemset Counting (DIC) forms the basis of our distributed algorithm, we would discuss this algorithm in detail.

DYNAMIC ITEMSET COUNTING (DIC)

Dynamic Itemset Counting (DIC) is an algorithm which reduces the number of passes made over the data while keeping the number of itemsets which are counted in any pass relatively low. In the first M transactions the algorithm starts counting the 1-itemsets. After M transactions for a given minimum support threshold, if any of the itemsets exceeds the minimum support threshold M transactions, then start counting the 2-itemsets count before waiting for a complete scan of database. In this way, DIC algorithm starts counting the 1-itemsets for involving data Count Distribution.The Data Distribution and Candidate Distribution shows that Count Distribution (CD) will perform Distribution as similar to Count Data Distribution. At the same time data is repartitioned so that a processor counts its candidate set independently. No communication of counts or data tuples are excepted for pruning the local candidate set. This information is sent asynchronously and the processors does not wait for complete information to arrive at that time. Each processor opportunistically starts counting the candidate sets using whatever information has arrived at other two level. We have compared our algorithm against CD. In these Data Distribution algorithm, each processor counts mutually exclusive candidates only. As the large number of processors are increased, a number of candidates can be counted in one pass. On other hand n-processor are configured using Data Distribution will be able to count in a single pass and candidate set that would require n passes in CD. The first pass of the algorithm is the same as CD where all the candidate 1-itemsets are counted. For all passes greater than 1, processor will act as a complete information. The drawback of this algorithm is that every processor must broadcast its local database to all other processors in every pass. The Candidate Distribution algorithm partitions the data and candidates in such a way that each processor may proceed independently.

STEPS OF DIC ALGORITHM:

Steps:

Step1. Mark the empty item set with a solid square. Mark all the 1-item sets with dashed circles. Leave all other item sets unmarked.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

Step2. While any dashed item sets remain:
Read M transactions (if we reach the end of the transaction file, continue from the beginning). For each transaction, increment the respective counters for the item sets that appear in the transaction and are marked with dashes.
If a dashed circle's count exceeds minsupp, turn it into a dashed square. If any immediate superset of it has all of its subsets as solid or dashed squares, add a new counter for it and make it a dashed circle.
Once a dashed item set has been counted through all the transactions, make it solid and stop counting it.
Explanation

Step 1: Mark the empty item set with a solid square. Mark all the 1-item sets with dashed circles. Leave all other item sets unmarked. Add new item sets. Maintain a counter for every item set. It is an algorithm which reduces the number of passes made over the data while keeping the number of item sets which are counted in any pass relatively low. Hence, the empty item set are marked as solid square while other 1-item sets are marked as dashed circles which indicate that the 1-item sets are suspected infrequent item sets which are counting below minimum support.

Step 2: While any dashed item sets remain:
Manage item set states from dashed to solid and from circle to square:

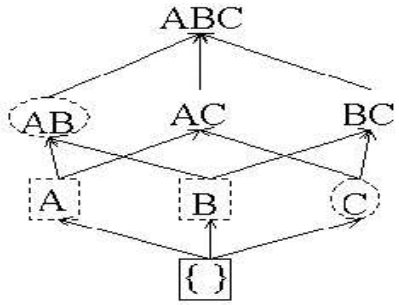
Read M transactions (if we reach the end of the transaction file, continue from the beginning). For each transaction, increment the respective counters for the item sets that appear in the transaction and are marked with dashes.

The M transactions are read first. For these each M transaction, a counter assigned to it is incremented when the item sets appear in the transactions and are marked with dashes. In the first M transactions the algorithm starts counting the 1- item sets. After M transactions for a given minimum support threshold, if any of the item sets exceeds the minimum support threshold in those M transactions, then we start counting the 2- item sets before waiting for a complete scan of the database. In this way, DIC starts counting the 1-item sets and then quickly adds counters for the 2,3,4,...k-item sets. We will define this M as a checkpoint. DIC uses checkpoints M transactions apart. DIC counts the frequent item sets and the minimal small item sets. Minimal small item sets are those item sets which form the boundary between the frequent item sets and the infrequent ones. Their subsets are frequent item sets.

If a dashed circle's count exceeds minsupp, turn it into a dashed square. If any immediate superset of it has all of its subsets as solid or dashed squares, add a new counter for it and make it a dashed circle.

If the dashed circle count exceeds minimum support, then this is replaced to a dashed square which indicated that the item sets are turned to suspect frequent item sets whose counting exceeds minimum support. And the immediate superset of such item set has all its subsets as conformed or suspects frequent items exceeding the minimum support then a counter is added to such item sets and made ad suspected infrequent item sets counting below minimum support. For every item set, the counting stops from the same point from where it started i.e. after one complete database pass. Thus an item set can be considered for counting at the next checkpoint instead of waiting until the end of the previous pass. Once a dashed item set has been counted through all the transactions, make it solid and stop counting it.

After all the dashed item sets are been counted in the transactions, all these item sets are turned to solid item sets and hence the counting is stopped as all the items in the transactions have been read When item sets become large determine which new item sets hould be added because they could potentially be large:



If the data is fairly homogeneous and for small values of M, DIC takes very few passes. If the data is no homogeneous or it is much correlated, it may not be realized that an item set is actually large until it has been counted in most of the database. This effect can be reduced considerably with randomizing the order of the transactions.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

D IC Example

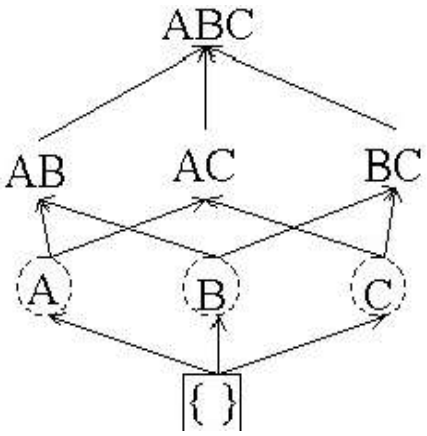
Min supp = 25% and M = 2.

ssTID	**A**	B	C
T1	**1**	1	0
T2	**1**	0	0
T3	**0**	1	1
T4	**0**	0	0

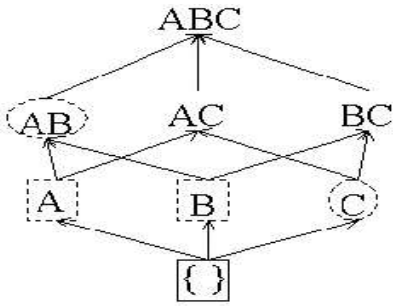
Table 1. Transaction Database

Item set lattice for the above transaction database

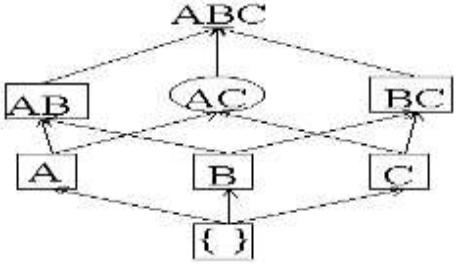
Item set lattice before any transactions are read:



In this Counter: a=0, B=0, C=0
Empty item set is marked with a solid box. All 1-item sets are marked with dashed circles.
After M transactions are read:



In this, Counters:A = 2, B = 1, C = 0, AB = 0
we change A and B to dashed boxes because their counters are greater than min sup (1) and add a counter for AB because both of its subsets are boxes.
After 2M transactions are read
In this, Counters: A = 2, B = 2, C = 1, AB = 0, AC = 0, BC = 0
C changes to a square because its counter is greater than min.sup. A, B and C have been counted all the way through so we stop counting them and make their boxes solid. Add counters for AC and BC because their subsets are all boxes.
After 3M transactions are read:
In this, Counters: A = 2, B = 2, C = 1, AB = 1, AC = 0, BC = 0
AB has been counted all the way through and its counter satisfies minsup so we change it to a solid box.
BC changes to a dashed box.
After 4M transactions are read:



In this, Counters: A = 2, B = 2, C = 1, AB = 1, AC = 0, BC = 1
AC and BC are counted all the way through. We do not count ABC because one of its subsets is a circle. There are no dashed item sets left so the algorithm is done.

HOW DIC WORKS:

Add new item sets.
Maintain a count for every item set.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

Manage itemset which states from dashed to solid and from circle to square.
When itemsets become large they determine which new item sets should be added because they could potentially be large dataset.
Aim of data mining is to extract important data from the database and it also provide reliability.

RELATED WORK

We are given a large database of customer transactions. Each transaction consists of items purchased by a customer in a visit. We present an efficient algorithm which generates all significant association rule mining algorithm between items in the large database.
This algorithm incorporates buffer management and novel estimation and pruning techniques. We also implement the results for applying algorithm to sale data in efficient way and obtained from a large company, which shows the effectiveness of the Apriori & DIC algorithm.

Mining frequent patterns in transaction databases, time-series databases, and many other kinds of databases has been studied popularly in data mining research. Most of the previous studies adopt an Apriori based candidate set generation-and-test approach. However, candidate set generation is still costly and expensive, especially when there exist prolific patterns and/or long patterns.

In this study, we proposed a novel called frequent pattern tree (FP-tree) structure, which is an extended prefix-tree structure for storing compressed, and data crucial information about frequent patterns, and develop an efficient FP-tree-based mining method. FP-growth for mining the complete set of frequent patterns know as fragment growth. Efficiency of mining is achieved by three techniques: a large database is compressed into a highly condensed, much smaller data structure, which avoids costly, repeated database scans. FP-tree-based mining adopts a pattern fragment growth method to avoid the costly generation of a large number of candidate sets,these partitioning-based, divide-and-conquer method is used to decompose these mining task into a set of smaller tasks. For mining task they confined patterns in conditional databases, which are dramatically reduces the search space. This performance study shows that the FP-growth method is efficient and scalable for mining both long and short frequent patterns. The order of magnitude faster than the Apriori algorithm and also faster than some recently reported new frequent pattern mining methods.

ADVANTAGES

Marketing / Retail

Through market basket analysis, a store can have an Data mining helps marketing companies build models based on historical data to predict who will respond to the new marketing campaigns such as direct mail, online marketing campaign…etc. Through the results, marketers will have appropriate approach to sell profitable products to targeted customer.Data mining brings a lot of benefits to retail companies in the same way as marketing. Appropriate production arrangement in a way that customers can buy frequent buying products together with pleasant. In addition, it also helps the retail companies offer certain discounts for particular products that will attract more customers.

Finance / Banking

Data mining gives financial institutions information about loan information and credit reporting. By building a model from historical customer's data, the bank and financial institution can determine good and bad loans. In addition, data mining helps banks detect fraudulent credit card transactions to protect credit card's owner.

Governments

Data mining helps government agency by digging and analyzing records of financial transaction to build patterns that can detect money laundering or criminal activities.
information available, the credit card stolen and identity theft become a big problem.

APPLICATIONS

For purchasing consumers buying habits: It can let us know the number of frequent items are bought by the consumers. In this way the buying habits of the consumer's way can think thatprocess are frequenct of subsets of data items are studied.

To study and implement various marketing strategies: It can be used to the various strategies for the implementation of various schemes.By these scheme we can consided the transactions of consumer's buying habits. This also help us to consided the historical transactions,and various marketing strategies are implemented.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

User's feedback: User's feedback can be included in future in the application for the betterment of the application. This application can be moderated as per the user feedback and this will help in making expert system more precise and user friendly.
Shopping Malls: Every Shopkeeper can see all the transaction and can also get idea about which frequent items are selling mostly.

Internet usages: We can keep record of mostly visited web sites in which people are using.
Fraud detection: In this fraud detection is not done because we have all transaction record.

CONCLUSION

In this paper, we have presented an algorithm called DIC is based on Dynamic Itemset Counting which represents a different approach for frequent itemset generation in a distributed DIC environment. We have observed and compared to CD, DIC shows much higher performance gain on sparse as well as dense datasets.

Thus we have successfully implemented algorithm based on DIC for large transactions, and thus provides the suitable techniques. The steps to study the frequent items as well association with data mining rules. To get the information about the transactions and we have successfully implemented DIC algorithm to obtain frequent item sets and generated item sets which are closely related to items and thus compared with respect to time complexity.

By comparing these, we conclude that Dynamic Item set Counting Algorithm is more reliable and efficient than Apriori Algorithm because Dynamic Item set Counting Algorithm require less passes as compare to Apriori algorithm.

REFERENCES

1. Assaf Schuster and Ran Wolff,Communication-Efficient Distributed Mining of Association Rules, ACM SIGMOD, Boston,4,May,2001.
2. Margaret Dunham, Data Mining, Introductory and Ad- vanced Topics, Pearson Education,2006.
3. Paranjape P,Deshpande U "An Optimistic Message Distibuted Algorithm For Association Rule Mining" India Conference (INDICON),2009Annual IEEE Digital Object Identifier :10.1109/INDCON.2009.5409349 Publication Year :2009.
4. Banu R.K, Ravanan R,GopalJ "Analysisand implementation of association rule mining"Signal and image Processing (ICSIP),2010 International Conference.
5. Yadav C,Shuliang Wang,Kumar M "An Approach to improve DIC Algorithm Based on Association Rule Mining"Computing ,Communication and Networking Techonology (ICCCNT),2013 Fourth International Conference On Digital Object Identifier 10.1109 (ICCCNT)2013.
6. Frequent Itemset Mining Dataset Repository, 2004 http://fimi.cs.helsinki. fi/data/.
7. R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rules between Conference,1993.
8. Z. K. Baker and V. K. Prasanna. Automatic Synthesis of E_cient Intrusion Detection Systems on FPGAs. In Proceedings of the 14th Annual International Converence on Field-Programmable Logic and Applications (FPL '04), 2004.
9. F. Bodon. A Fast Apriori Implementation. In Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining implementation 2003.
10. M. Estlick, M. Leeser, J. Szymanski, and J. Theiler. Algorithmic Hardware.In Proceedings of the Ninth Annual IEEE Symposium on Field Programmable Custom Computing Machines 2001 (FCCM 01), 2001
11. K. Gaber, M.J. Bahi, and T. El-Ghazawi. Parallel Mining of Association Rule with a Hop_eld-type Neural Network. In Proceedings of Tools with Artiifcial Intelligence (ICTAI 2000),
12. E. (Sam) Han, G. Karypis, and V. Kumar. Min-Apriori: An Algorithm for Finding Association Rules in Data with Continuous Attributes, 1997.
13. E.(Sam) Han, G. Karypis, and V. Kumar. Scalable Parallel Data Mining for Association Rules.IEEE Transactions on Knowledg and Data Engineering.
14. The Xilinx Corporation. ML-300 Development Board.
15. The Xilinx Corporation. Virtex II Pro Series FPGA Devices.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

16. 20. C. Wolinski, M. Gokhale, and K. McCabe. A Recon_gurabrage Computing Proceedings of the ,2004.
17. R. Agrawal , T. Imielinski , and A. Swami. Mining association rules between sets of items in large databases. In Proc. 1993 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'93), pages 207–216,Washington, DC, May 1993
18. S. Brin, R. Motwani, J.D. Ullman, S. Tsur, "Dynamic Item set Counting and Implication Rules for Market Basket Data", SIGMOD Record, Volume 6, Number 2: New York, June 1997, pp. 255 - 264.
19. Su, Yibin, Dynamic Item set Counting and Implication Rules for Market Basket Data: Project Final Report, CS831, April 2000