Incremental Frequent Pattern Mining

Jyoti Jadhav, Lata Ragha, Vijay Katkar

Abstract*—Association rule discovery is widely used Data Mining technique for Market Basket Analysis. It discovers interesting correlations and frequent patterns from the database. In real life, new transactions are continuously added to the database as time advances. This result in; periodic change in correlations and frequent patterns present in database. Incremental Association Rule mining is used to handle this situation. Most of the existing Incremental rule mining methods are highly dependent on availability of main memory. If sufficient amount of main memory is not available, they fail to generate the results. This paper presents a novel method for incremental discovery of frequent patterns using Main Memory database Management System to eliminate this drawback. Experimental results are provided to support the efficiency of proposed method.

***Index Terms*— Apriori, FP-tree, Incremental Association Rule Mining, Main memory database Management System**

I. INTRODUCTION

Data Mining [1] [2] [3] is nothing but efficient discovery of unknown pattern from the large database or non trivial extraction of useful information from large set of database. Data Mining is one of the important step of knowledge discovery in database (KDD). KDD [4] refers to the process of identifying novel as well as important data from huge database. Data mining has emerged in the areas such as Banking (loan/credit card approval predict good customers based on old customers), Customer relationship management (identify those who are likely to leave for a competitor), Targeted marketing (identify likely responders to promotions), Fraud detection (telecommunications, financial transactions) etc. It consists of data which can grow and shrink when any insertion, deletion or modification is done.

There are different techniques for data mining. Predictive mining is the technique which helps to predict unknown variables or future values of other variables and Descriptive mining is technique which helps to find human-interpretable patterns that describe the data. Other tasks are such as Classification, Regression and Deviation (Predictive mining), Clustering, Association Rule Discovery, Sequential Pattern Discovery (Descriptive mining).

The most important task in association rule mining is; find out association, frequent patterns, correlation among the existing itemsets from given database. In given set of transaction, where transaction consists of items such as X and Y then association Rules are denoted as X=>Y and intersection between them is null. The support of this rule is defined by percentage of transaction that contain set X and Confidence of the rule is defined as percentage of these X transaction that also contain Y. To generate frequent items certain user defined support is called as minimum support. In association rule mining, the item having support is greater than its minimum support is called as frequent item. The algorithm such as apriori and FP-tree based on association rule mining are most useful to discover frequent pattern.

Many researchers have proposed several algorithms to discover frequent pattern but most of them are static in nature. In real world data is growing continuously. The static algorithms are not able to work efficiently whenever any change happens to the original database. Existing support as well as association rule may invalidate. If support is static (once it set to very high or low) important information loss may occur. Support should be flexible, based on real time event to avoid loss of important information.

One solution to this problem is to re-apply the algorithm on new database (original + new) but in this case time and CPU utilization is very high. This approach is costly whenever small amount of data is inserted. The advantage of incremental Mining is to process only incremented part and to discover frequent patterns. Efficiency of these algorithms is based on number of passes as well as scans required for processing and number of candidate set are generated. The large set of evolving and distributed data can be handled efficiently by Incremental Data Mining.

This paper presents a new incremental algorithm has introduced which is incremental in nature. The goal is to discover frequent items whenever new data is added dynamically to the original database. This approach introduces without pruning infrequent because that can become frequent after inserting new transactions to the original database. It also reduces number of passes for processing and scans to the original database. This may result that execution time is faster than previous methods with help of parallel processing.

Remainder of this paper is organized as follows. Section 2 gives the brief overview of recent work done in Frequent Pattern Mining (FPM). Section 3 presents comparison of recent FPM methods. Section 4 describes new proposed mechanism. Section 5 gives performance evaluation followed by conclusion in section 6.

II. RELATED WORK

Apriori Algorithm [3] [5] is one of simplest approach to generate frequent pattern. This algorithm is recursive in nature, so processing is iterative (brute force approach). In first iteration candidate-set of size-1 (C1) is generated, and then whole database scanning is done. The items having support greater than user defined minimum support are used as frequent items (L1) of size-1.

Manuscript received on August 25, 2012

Miss. Jyoti Jadhav, Computer Department, Mumbai/ R.A.I.T/ DY. Patil, Navi Mumbai, India.

Dr. Lata Ragha, Computer Department, Mumbai/ Terna Engineering College, Navi Mumbai, India.

Mr. Vijay Katkar, His Department Name, Pune/ PCCOE, Pune, India..

This process continuously till Ci or Li becomes empty. It is basically candidate-set generation and test approach. Disadvantages of this is that large number of candidate generation and time consuming as it required multiple passes for processing.

FP-tree [6] [7] [8] is one of best approach to discover frequent pattern to overcome the drawback of the apriori algorithm. It requires only two passes of processing. One pass is required for ordering and structuring frequent items other pass is for inserting those frequent items in the tree. FP-tree as better performance than Apriori as reduce database scan. Since even if small insertion is done, restructuring of item is required again to arrange in descending order. FP-growth [8] [9] algorithm is applied on FP-tree to discover frequent pattern. It is based on divide-conquer approach to discover frequent pattern of various sizes.

G. Pradeepini and S. Jyothi [10] have proposed algorithm called Tree-based incremental Association rule mining (TIARM) algorithm. This algorithm has two different mechanisms. First, is to generate INC-tree which is more enhanced than FP-tree to make tree more compact in nature. Second, TIARM is applied on INC-tree to discover frequent patterns of different sizes. The process of generating INC-tree is same as that of the FP-tree with single pass processing. By using conditional pattern base and FP-tree, frequent patterns are generated without candidate itemset.

Liu Jian-ping et al [11] present an algorithm called FUFP-tree based incremental association rule mining algorithm (Pre-FP). It is based FUFP [12] [14] (Fast Updated Frequent Pattern) concept. The major idea of FUFP is re-use of previously mine frequent items to update with incremental database. It reduces number of candidate set in updating process. All the links in FUFP are bidirectional where in FP-tree links are only in single direction. Advantage of bidirectional link is easy to add remove child node without much reconstruction. This FUFP structure is used as input to the Pre-large, which gives positive count difference whenever small amount of data is added to original database. It deals with change in database in case of inserting new transaction. The algorithm classify items into the three categories: Frequent, infrequent and pre-large. Pre-Large [13] itemsets has two support threshold values i.e. upper and lower threshold. These support thresholds are helpful for maintaining cost while insertion and deletion of items into original dataset. These items are differentiated between nine cases in first pass. Each case is handled efficiently to discover frequent pattern in second pass. Such type of characteristics is useful for real-world applications such web mining.

Chowdhury Farhan Ahmed et al. [18] have proposed two Single-pass incremental and interactive frequent itemsets mining algorithms with single database scan. One is weight in ascending order (i.e. IWFPwa) in which each item is having specific weight (different degree of importance). In this algorithm the given weight of items are used to calculate support of items in the database. Those weights are sorted in ascending order with highest weight in bottom this leads to database size reduction. This compressed structure is used to build FP-tree and then FP-growth algorithm is applied to discover frequent pattern. Another algorithm is based on frequency by arranging it in descending order (i.e. IWFPfd). The main advantage of this algorithm is prefix sharing of node [19] with compact structure of the tree. Numbers of nodes are less as compared to the previous method which saves memory space.

Siqing Shan et al. [15] have presented Incremental Association Rules Mining method based on Continuous Incremental Updating Technique. Transaction Amalgation Algorithm is used to merge the transaction in transaction database based on quantity present in transaction in descending order. That reduces the overall size of the database drastically saves memory space. T-tree algorithm is applied on these database which works as FP-tree. Finally T-tree is given as input to the FP-growth algorithm to discover frequent pattern. Each pattern in overall database (original+ new) is applied to candidate pattern pool, where it is classified in four cases:

i. Pattern may frequent in old database and not frequent loser in increment to database
ii. Frequent in both old database as well as increment to it
iii. Not frequent in both old database as well as increment to it
iv. Frequent in increment to database and not frequent in old database

D. Kerana Hanirex and Dr. M. A. Dorai Rangaswamy [20] have proposed clustering based incremental algorithm to discover Frequent Patterns. The partitioning algorithm has proposed to generate cluster. Then Improved Apriori Algorithm [21] is applied to generate frequent patterns. If pattern is frequent then it is present in any of the cluster. Whenever new transaction is added to the database it treated as new cluster. Again Improved Apriori algorithm is applied to discover newly frequent pattern in incremental database. This algorithm has better efficiency than previous Apriori algorithm by reducing memory space and number of passes.

Liu Han-bing, Zhang Ya-juan, Zheng Quan-lu and Ye Mao-gong [22] has proposed Incremental Frequent Pattern mining algorithm based on AprioriTidList Algorithm [23]. This algorithm also improves Apriori performance by pruning transaction. It requires only one database scan which make it more efficient. It scans a database and creates a Tid List .It does not uses whole database to count support value instead it consider particular large item in transaction with identifier TID. If transaction does not contain that large item then that transaction is deleted which reduces database size drastically. Tid list of Item ‘I’ contain list of all the transaction in which I is present. Tid list of Item ‘J’ contains list of all the transactions in which J is present. Intersection of both the list gives the list of transaction in which both I and J are present. When new data is added it discover frequent pattern using old frequent pattern.

Shih-Sheng Chen et al. [16] have proposed a method for discovery of frequent periodic pattern using multiple minimum supports. This very efficient approach to find frequent pattern because it is based on multiple minimum support based on real time event. All the items in the transactions are arranged according to their MIS (Minimum Item Support). It does not hold downward closure property instead it uses sorted closure property based on ascending order. Then it uses PFP [17] (Periodic Frequent Pattern) whose construction is same as that of the FP-tree. Finally, PFP-growth algorithm is applied which is same as that FP-growth and conditional pattern base is used to discover frequent pattern. This algorithm is more efficient in terms of memory space and database scan by reducing number of candidate set.

III. PROPOSED METHOD

This paper presents a parallel frequent pattern generation mechanism using MMDBMS. It is divided into two types of modules; Central Co-ordinator module and Parallel processor module. Central co-ordinator module divides the available Dataset or increment to Dataset into N parts using horizontal partitions; here N is number of available processors. Parallel processors perform their task to generate frequent patterns and return the result to central co-ordinator. Finally Central co-ordinator combines all the results.

A. Working of Central Co-ordinator

Input: Dataset or Increment to dataset; say D
No of available Processors; say N
Output: Frequent Patterns
Algorithm:
Step 1: Divide D into N parts using horizontal partitioning
Step 2: Send one partition of D to every processor
Step 3: Collect result (in the form of MMDBMS database) from every processor
Step 4: Generate final result (by combining tables of all databases)

B. Working of Parallel Processor

Input: Dataset or Increment to dataset sent by Central Co-ordinator; say CD
Output: Frequent Patterns
Algorithm:
Step 1: for every transaction Ti in CD

a) Create a table in MMDBMS
b) Sort items of Ti in ascending order
c) Generate patterns of size-1, size-2, … size-n; where n is size of transaction Ti
d) Update frequency count of every generated pattern in MMDBMS table

Step 2: Send MMDBMS database (containing table) to Central Co-ordinating process

C. Working Mechanism in Single Processor system

Working mechanism of proposed method in single processor environment is as described below:

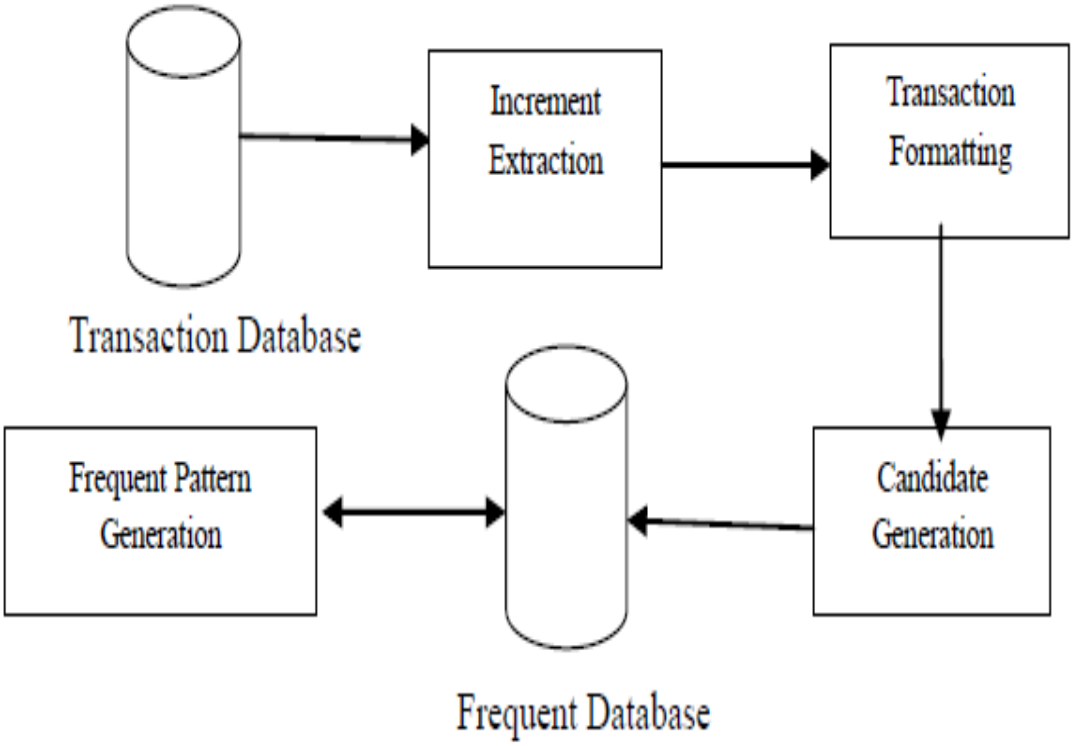

Figure1: Proposed Method

Working of each module:

1. **Transaction Database**: Database which contains day to day transactions.
2. **Increment extraction**: As transaction database is incremental in nature some addition, deletion or modification occurs on to it. It extracts those changes.
3. **Transaction Formatting:** Items in transactions are arranged in ascending order for processing.
4. **Candidate item sets** of different size are generated.
5. **Frequent Pattern Database:** Contains occurrence frequency of each itemset.
6. **Frequent patterns** are generated by taking consideration of candidate generation with their count values in given database.

IV. COMPARISON

Table 1 summarizes the comparison of proposed method with existing approaches.

TABLE I: Comparison of proposed method with existing approaches

Method	No. Of Database Scans Required	No. of Passes Required	Approach used	Remove less frequent itemsets of Size 1	Relative CPU Utilization	Dependency on Main Memory	Parallel Processing
TIARM	1	Multiple	Divide and conquer	Yes	High	High	No
Pre-FP Based FUFP	2	Multiple	Pre-large itemsets(Lower and upper threshold)	Yes	Medium	High	No
CIU	2	Multiple	Transaction merging	Yes	Medium	High	No
Weighted frequent pattern	1	Multiple	Tree Restructuring	Yes	Medium	High	No
Cluster based Apriori	Multiple	Multiple	Clustering	Yes	High	High	No
Incremental Updating using AprioriTidList	1	Multiple	Intersection of Link List	No	Low	High	No
PFP-tree	2	Multiple	Divide and conquer	Yes	Medium	High	No
Proposed Method	1	Single	Counting Based	No	Low	Low	Yes

All algorithms except proposed method are highly dependent on the main memory which makes them less efficient. Apriori-Tid uses link list structure; which may not fit into main memory if size of database is huge. Most of other methods use FP-tree, memory requirement for FP tree is very high. In cluster based algorithm it as assumption that cluster are made such a way that it will fit into main memory, which is not possible in case of very huge databases.

TIARM, Weighted pattern, Pre-FP, CIU and PFP-tree scans the database only twice but they requires multiple passes of data processing to discover Frequent Patterns. This multi pass processing increases overhead in terms of the Time and CPU requirements. All above algorithm prunes infrequent items of size-1 after and before creation of tree. It may leads to the reduction of CPU requirement during processing of incremental data.

Major limitation of all previously discussed algorithms is; they do not support parallel processing as FP-tree and link list cannot be implemented using distributed processing. Proposed method can be implemented in distributed manner which increases the efficiency.

V. EXPERIMENTAL RESULTS

All experiments are performed on Intel core i_5 processor 2.40 GHZ with turbo Boost up to 3.00GHZ with windows 7 operating system and 4 GB RAM for single processing environment. In case of parallel processing two or three processor are connected in local Area Network (LAN).

The given dataset is distributed among the number of available processor by Co-ordinator having the same configuration or it can run on i_3 processor with 2 GB RAM. Configuration may change depends on total size of incremented dataset. Finally, results are calculated by each processor are combined by Co-ordinator to discover final frequent patterns with their different counts. Dataset are used for experiment is mushroom or chess dataset. JDK 1.6 is used as programming language. This paper uses HSQLDB as MMDBMS. HSQLDB is more efficient than SQL relational database engine written in Java having additional feature. It offers a small, fast multithreaded and transactional database engine with in-memory and disk-based tables. Processor(s) can directly access Main memory thus access time for data stored in it is much lesser compared to data stored in secondary memory (disks). With increase in densities of semiconductor chips and decrease in cost, it is feasible to store huge amount of data in main memory [24]. MMDBMS uses main memory as primary storage for huge amount of data and provides very high speed access to it. This makes it suitable option for implementation of real-time network security systems [25].

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

A. *Performance Analysis*

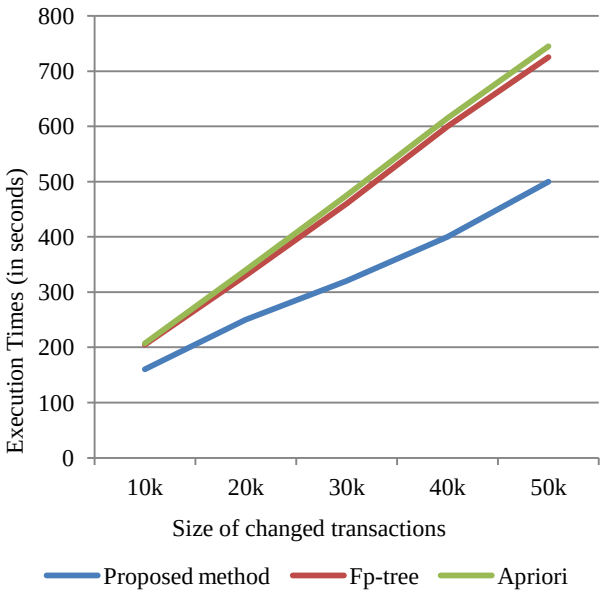

Figure 2. Scalability with respect to increase in number of transaction

From, above performance analysis it can be seen that proposed method is more efficient than FP-tree and Apriori. Most of existing algorithms are based on basic FP-tree and Apriori. As the numbers of transactions are incremented in to original database, time is required for processing is less as compared that both because proposed method support for parallel processing.

VI. CONCLUSION

In this paper, new method is proposed for finding frequent pattern in case of incremental database. As this method takes only one database scan and pass for processing frequent pattern. It is based on concept called as MMDBMS with help of HSQLDB. It also works efficiently in single as well as multi processing environment which gives better and faster performance than other existing algorithms.

REFERENCES

1. Ruowu Zhong, Huiping Wang,"Research of Commonly Used Association Rules Mining Algorithm in Data Mining", International Conference on Internet Computing and Information Services, pp. 219 – 222, IEEE 2011.
2. R. Agrawal, T. Imielinski, and A.N. Swami "Mining association rules between sets of items in large databases," Proceedings of ACM SIGMOD International Conference on Management of Data, ACM Press, Washington DC, pp.207-216, May 1993.
3. R. Agrawal, and R. Srikant, "Fast algorithms for mining association rules," Proceedings of 20th International Conference on Very Large Data Bases, Morgan Kaufmann, pp. 487-499, 1994.
4. Agrawal, R. and Psaila, G. Active Data Mining "Proceeding of 1st international Conference knowledge discovery and database", Montreal, Auguest 1995.
5. Agrawal R.; J.C Shaifer Parallel Mining of Association Rule " IEEE transaction on Knowledge and Data Engineering,Vol.6, Dec 1996.
6. "Improvement and Realization of Association Rules Mining Algorithm Based on FP-tree", Vol.4, pp.463, IEEE 2010.
7. Jiawei Han, Jian Pei, Yiwen Yin, Runying Mao, "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree", Data Mining and Knowledge Discovery, pp.53–87, IEEE 2004.
8. J. Han, J. Pei and Y. Yin, "Mining frequent patterns without candidate generation," The ACM SIGMOD International Conference on Management of Data, pp. 1-12, 2000
9. Huanglin Zeng Sichuan and Bangrong, "An Improved Algorithm of FP - tree Growth Based on Mapping", International Conference on Computer Application and System Modeling (ICCASM), 2010.
10. G. Pradeepini, S. Jyothi, "Tree-Based Incremental Association Rule Mining without Candidate Itemset Generation", Trendz in Information Sciences & Computing (TISC), pp. 78-81, IEEE 2010.
11. Liu Jian-ping, Wang Ying, Yang Fan-ding, "Incremental-Mining algorithm Pre-FP in association rules based on FP-tree", Networking and Distributed Computing(ICNDC), First international Conference, pp.199-203, IEEE 2010.
12. Lin, C.-W., Hong, T. –P., & Lu, W. –H. (2009). "The Pre-FUFP algorithm for incremental mining" Expert Systems with Applications, 36, 9498–9505.
13. T. P. Hong and C. Y. Wang, "Maintenance of association rules using pre-larg itemsets," Intelligent Databases: Technologies and Applications, Z. Ma (Ed.), Idea Group Inc., pp. 44-60, 2006.
14. T. P. Hong, J. W. Lin, and Y. L. Wu, "Maintenance of fast updated frequent pattern trees for record modification", The International Conference on Innovative Computing, Information and Control, pp. 570-573, IEEE 2006.
15. Siqing Shan, Xiaojing Wang, and Miao Sui "Mining Association Rules: A Continuous Incremental Updating Technique", 39In proceeding of: Web Information Systems and Mining (WISM), International Conference on, Volume: 1, pp. 62 – 66, IEEE 2010.
16. Shih-Sheng Chen, Tony Cheng-Kui Huang, Zhe-Min Lin, "New and efficient knowledge discovery of partial periodic patterns with multiple minimum supports", The Journal of Systems and Software 84, pp. 1638– 1651, ELSEVIER 2011.
17. Gu, C.-K., Dong, X.-L. "Efficient mining of local frequent periodic patterns in time series database", International Conference on Machine Learning and Cybernetic, pp. 183–186, 2009.
18. Chowdhury Farhan Ahmed, Syed Khairuzzaman Tanbeer, Byeong-Soo Jeong, Young-Koo Lee a,Ho-Jin Choi(2012) "Single-pass incremental and interactive mining for weighted frequent patterns", Expert Systems with Applications 39 pp.7976–7994, ELSEVIER 2012.
19. Tanbeer, S. K., Ahmed, C. F., Jeong, B.-S., Lee, Y.-K., Efficient single-pass frequent pattern mining using a prefix-tree. Information Sciences, 179(5), pp.559–583, 2009.
20. D.Kerana Hanirex, Dr.M.A.Dorai Rangaswamy "Efficient Algorithm For Mining Frequent Itemsets Using Clustering Techniques", International Journal on Computer Science and Engineering (IJCSE), Vol. 3 No. 3 Mar 2011.
21. Sheng Chai, Jia Yang and Yang Cheng, "The Research of Improved Apriori Algorithm for Mining Association Rules" In Proceedings of the Service Systems and Service Management, International Conference, 9-11 June 2007 pages : 1 – 4,2007.
22. Liu Han-bing, Zhang Ya-juan, Zheng Quan-lu, Ye Mao-gong, "New Incremental Updating Algorithm for Mining Association Rules Based on AprioriTidList Algorithm", Cross Strait Quad-Regional Radio Science and Wireless Technology Conference (CSQRWC), Vol. 2, pp 1611 – 1614, IEEE 2011.
23. Kotasek Peter and Zendulka Jaroslav, "Comparison of Three Mining Algorithms for Association Rules", Proceeding of 34th Springer International Conference on Modelling and Simulation of Systems (MOSIS'2000).
24. Hector Garcia-Molina, Kenneth Salem, "Main Memory Database Systems: An Overview", IEEE Transactions on Knowledge And Data Engineering, Vol. 4, No. 6, December 1992.
25. F. Raja, M. Rahgozar, N. Razavi, and M. Siadaty, "A Comparative Study of Main Memory Databases and Disk-Resident Databases", World Academy of Science, Engineering and Technology, Issue 14, 2006.

AUTHORS PROFILE

Miss. Jyoti Jadhav, 2nd year M.E.CSE (pursuing) B.E. Computer Science from Mumbai University. Two papers published in two international conferences. Area of interest Data Mining ,Security, Image Processing.

Dr. Lata Ragha, completed Phd in 2011. Papers published in international conference and journal around 50 papers. Currently working in Terna Enginnering College in Mumbai University.

Mr. Vijay Katkar, completed M.Tech from VJTI in 2008. Currently working in PCCOE and pursuing PHD from Bharti Vidhyapith, Pune. Total 12 numbers of papers published in international conferences and journals.