Design and Analysis of a Novel Approach for Finding Minimum Number of Representative Pattern Sets

Raju Sunkari [#1], **M.Kranthi Kumar** [*2], **Ramesh Challagundla** [*3]
M.Tech Scholar [#1] *, Assistant Professor* [*2] *, Professor & Principal*[*3]
Department of Computer Science & Engineering,
Pydah College of Engineering and Technology, Gambheeram,
Visakhapatnam, AP, India.

Abstract: **In current day's data mining has become one of the fascinating domains in which each and every organization use this in some or some other way to process their raw data. It is also described as the computational process of discovering patterns in large data involving methods at the intersection of artificial intelligence , ML approach, statistics , and database systems. The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. Generally there is a concept called as pattern mining where this is used for mining a large volume of data in order to find the frequent item sets as well as order of frequency. Also this FP Mining is very difficult to understand and further analysis of generated patterns is complex. As this is a complex task now we need to find out the smallest set of representative patterns to defend all other patterns. In this paper we have used an algorithm to find the minimum pattern set called as MINRPSet. By using this algorithm we can able to retrieve the patterns efficiently from a tree of nodes. All the patterns generated after applying the constraint may have a set cover problem; greedy set cover algorithm is used to avoid the set cover problem. By conducting various experiments on our proposed algorithm, we finally came to an conclusion that the greedy algorithm gives the minimum number of representative pattern sets.**

Keywords: **Pattern Mining, Frequent Mining, Greedy Algorithm, Data Mining, Machine Learning Approach.**

1. Introduction

Data mining is the process of extracting useful or structured information from a raw or un-structured data. Generally this is used mainly in performing operations like insurance sector, bank and retail sector, hospital for identifying diseases, shopping malls to calculate the priority of items that were sold. Data mining is the evaluation connected with info regarding interactions who have not necessarily previously recently been found. One example is, the actual gross sales documents for the distinct model of tennis racket may possibly, in case adequately reviewed as well as linked to different industry info, show the temporary. It is very challenging to comprehend or perhaps further evaluation of that massive data bank. On that basis the very idea of recurrent style mining is usually released. Pattern: the style is usually a collection of products, subsequences or perhaps substructures, what we are trying to find. Natural meats get a large number of styles in our data bank. Between the many styles we are meant to discover the many recurrent styles. The best way to quarry recurrent styles: the style that develops generally in a dataset is really a recurrent style; we've many algorithms to quarry recurrent styles just like apriori criteria, FP- increase criteria.

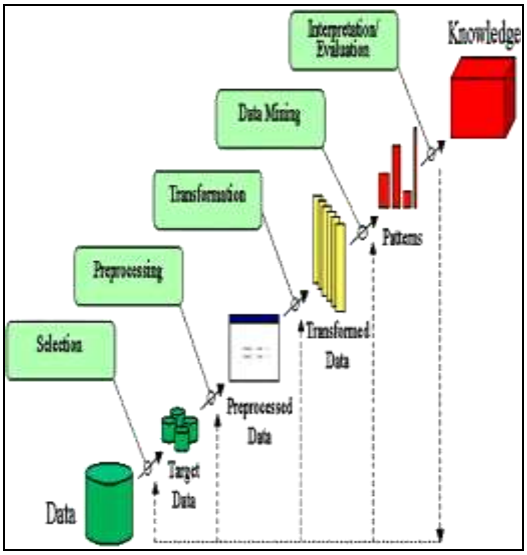
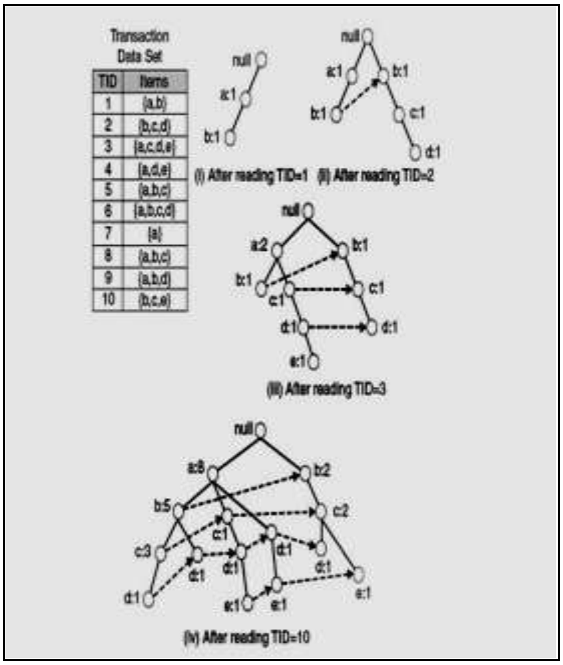


Figure. 1. Represents the Architecture of a Data Mining

From the figure 1, we can clearly find out that mining is a small part in the process of knowledge discovery.Initailly the data set will be pre-processed and cleaned in order to reduce the redundant items or ir-relevant items that are available in the data sets, once the data set is cleaned and pre-processed then we can able to send for the next process like classification and in turn for clustering and rule mining by using various data mining algorithms. One among the several algorithms in classification is Fp-miner where this is used for clustering the frequent item sets from a set of distinct elements.

Fp- growth is an formula regarding frequent pattern set exploration in addition to except apriori formula this specific formula permits frequent pattern set development with no customer piece set technology. A couple move methods: 1: develop a lightweight data construction named the actual fp-tree. a couple of: components frequent pattern packages specifically from your fp-tree.


Transaction Data Set

TID	Items
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}

Figure. 2. Represents the Sample Architecture of a FP-Growth Tree

From the above figure 2, we can clearly get an idea that clearly idea regarding Fp-Growth Tree in the form of TID = 1, 2, 3,4…n.Where 'n' is the highest frequent pattern number for mining the item sets. In the FP-Growth algorithm initially we will mine the data set and try to find 1-pattern sets those which are having similar matching in the overall data set, then the process continues for 2-set mining where we will try to match 2 item sets which are having similar features and so on this set continues till we find the highest level of pattern sets. Once the highest level of pattern matching is achieved then the process stops at that level. As the algorithm starts at level 1 and continues till 'N' level this will be represented in the form of a tree, hence we called the algorithm as FP-Tree. The detailed explanation of this FP-Tree algorithm is explained in the later sections.

2. Background Knowledge

In this section we will find the related work that was analyzed and studied in order to implement this current paper. This section will describe the work related to mining frequent item sets from a large data set.

2.1 Preliminary Knowledge

In data mining area, FP mining is very important problem. It was introduced by agarwal et al., in 1993. Usually FP mining is executed on transactional database

D ={t1,t2,----tn},

Where tj is a transaction which contains a set of items, **J$\in$ [1, n]**.

Let us assume **I= {I1,i2,---im}** are the set of trenchant items, all these items are appearing in D and X is a pattern it appears in the items of I. i.e **X$\subset$ I**.

If a transaction $T \in D$ holds all the item of a pattern **X**, then **T** is a supporting transaction of **X** and **T** supports **X**.

Assume **T(X)** is a set of transactions, which supports pattern **X**, then the support of **X** is denoted as sup(X). And if the support of a pattern **X** is bigger than the user defined minimum threshold then that pattern is called frequent pattern. Mining is performed to find all the FP's which are in the database D, whose support is more than the user specified threshold. There are many algorithms developed for mining FP's now the focus is shifted to how to mine the patterns efficiently and how to utilize them efficiently. All the FP's have the anti monotone property. This property implies if a pattern is frequent then all of its subset patterns must be frequent too. In a simple way, if a pattern is not frequent then all of its super sets are also not frequent. After finding the complete set of FP's, they may have lot of redundancy. Many FP's have similar items and similar supporting transactions.

If the grouping if all the similar items have been done. Then the number of FP's reduces. For this reason and for grouping, the concept of FP closed pattern proposed a pattern X is closed if it is more frequent than its sub items. The subsets of a closed pattern are also frequent items. From the closed patterns, the entire pattern and their support can be recovered. For finding the minimum representative pattern sets, two algorithms have been proposed, **MINRPSet** and FLEXRP set, these uses a tree structure called CFP- tree. CFP- tree (condensed frequent pattern sets) structure supports the efficient retrieval and structure of patterns. **MINRPSet** is some order of magnitudes slower than the database is FLEXRP set. FLEXRP set uses one extra parameter K to cover a pattern minimum number of times. But when the size of K is too large the FLEXRP set produces more number of representative pattern sets and when K =1 it gives faster access then **MINRPSet**.

Several approaches to develop algorithms for mining FP's, these are classified into two types:

1) Apriori family algorithms
2) Pattern -growth based algorithms

Apriori family algorithms are iterative algorithms. These algorithms use a transactional database and minimum support threshold as an input and then patterns larger than the min – support are as output. These algorithms assume that the datasets are noise free and well preprocessed. But in reality, datasets contains noisy values, mining values and somewhat dirty. So it is difficult to user to set minimum threshold value to get needed result. If the given min support is too large, then there may be a smaller number of FP's and these are not the desirable result. If the given min- support is too small there may be too many redundancy un-useful FP's. This takes very large processing time for mining and also the complexity is increase to filter all the un- interesting patterns.

Apriori algorithms works in two levels, in the first level it yields all the possible item set combinations. We call these combinations as candidate sets, these candidates are the inputs of other phase. The minimum support is applied in second phase, this support is applied to find all the FP's in the dataset. The rules in apriori algorithm are formed then rules are formed by using frequent item sets, and the minimum confidence constraint. The draw base of an apriori algorithm is that it generates a very large set of candidates.

FP growth based algorithms overcomes the drawback of apriori algorithm. These algorithms are implemented without generation of candidate sets. For mining the FP's, this algorithm uses a FP- tree structure, a frequent pattern (FP) tree structure. All the information collected from the data set are stored in the FP- tree. First of all these algorithms scans the dataset only ones and collects all the FP's and their supports, then sorts these FP's by their descending order based on the support. There is no candidate set generation in the FP tree. So it is more efficient than apriori. The FP-tree, traverse the tree by using the depth first search strategy. It means the FP's based on conditional patterns and creates child FP tree. This algorithm gives an efficient retrieval of FP's.

2.2 RPLocal and RPGlobal Algorithms

RP local and RP global are the greedy methods developed by X in. RP local is close to RP Global, these two algorithms can reduce the number of closed FP's. There are two steps RP global patterns, in the first step, it finds all the frequent patterns in the database. In the second step, it finds the set-covers to find the set of representative patterns.

In RP local, each pattern is scanned twice using depth – first searcher. In the first scan, each child visits from its parent and in the second scan, visit after the calls to its child when the pattern is visited second time, all the pattern that can cover it have been counted, all the other pattern are not able to cover it. RP local scans the output patterns, during the scanning, when the uncovered FP found then the RP local algorithm finds the larger set that in a representative pattern. This representative pattern is a largest set.

FP – growth method developed by han et al. Mines complete set of frequent item sets without generating candidates. It uses a concept of divide and conquers, it scans the database first. The list of frequent items are generated after scanning and all the frequent items are ordered according to their descending order, finally the descending ordered frequency items are compressed into a tree called FP tree. A pattern X is called maximal FP in dataset D if the pattern is frequent and there does not exist super pattern Y. Such that X $\subset$ Y and the pattern Y is frequent in D.

A pattern X is called closed pattern in dataset D if the pattern is more frequent than all of its super sets of a pattern. The maximal patterns contains all the frequent patterns and it does not holds the support but the closed patterns do. Prefix span is an a pattern growth technique for mining sequential patterns. It stands for prefix – projected sequential pattern mining. It is developed by jian pei. This implementation mines the patterns without generating the candidate sets or candidate subsequences. This gives the better efficiency and fast computation compared to apriori. If an item is infrequent than a projected item set of any sequence which is superset of that item set cannot be a sequential pattern. That property is called as free span property. It reduces the candidate generation. This technique mines all the patterns by partitioning the search space and based on the projected item sets.

To further reduce the size of the pattern sets, many concepts have been proposed. Such as disjunction – free generators, generators, δ – free sets, maximal patterns, redundancy aware top-K patterns, non- desirable patterns, top –K frequent closed patterns are proposed. In some datasets the non-desirable item sets are larger than the closed patterns/item sets. The total numbers of closed patterns are more than the total number of maximal patterns. Maximal patterns cannot able to recover their support but it can recover all the FP's.

The main problem in FP mining is that it requires scanning the dataset repeatedly and generation of candidate sets. The FP – growth algorithm is somewhat efficient then apriori. RP global is time consuming and it produces less representative patterns. RP local is efficient but it produces more number of representative patterns then RP global for finding frequent representative patterns, two algorithms **MINRPSet** and FLEXRP set have been developed. Both mines the FP's first and then finds the RP pattern sets. Both algorithms use the CFP – tree structure to store and retrieve the FP's.

Recurrent style mining is a time-consuming procedure primarily pertaining to very large datasets. It is therefore greater to employ a tactic "mining after as well as employing quite a few times". A main obstacle even though mining frequent styles from large data collection is usually that will this sort of mining builds a lot of styles through contemplating min_sup patience. This kind of happens when min_sup is determined minimal if a new style is usually frequent after that its subpatterns are also frequent. A sizable style includes rapid quantity of smaller sized as well as frequent sub-patterns. This issue is usually overcomed by using ideas like finished frequent style mining as well as maximal frequent style mining.

3. A CFP-Tree Structure

A compact disk- based structure for storing and querying frequent item sets. This tree is completely different from the other trees we have already known. When the minimum support threshold is low and the dataset is huge, then the frequent patterns generated may be large. Those patterns are huge to store in any tree, so that the CFP tree has been introduced. It uses both suffix sharing and as well as prefix sharing among items so that the space can be saved. It is not desired to access the data from dataset, because the data can be accessed from the CFP – tree directly itself. The size problem in this tree is totally minimized, and it supports efficient retrieval of pattern / frequent items.

TID	Transactions
1	a, c, e, f, m, p
2	b, e, v
3	a, b, f, m, p
4	d, e, f, h, p
5	a, c, d, m, v
6	a, c, h, m, s
7	a, f, m, p, u
8	a, b, d, f, g

Table 1: Represents the Sample Dataset D

The CFP – tree structure is same as the prefix tree structure. Each and every node in this tree is variable – length array and all the entries in the node are stored in ascending order means ascending frequency order. When we were comparing the CFP – tree structure with FP – tree structure, it is completely different by several aspects. FP – tree stores all the projected transactions, when the mining is goes on. Coming to CFP – tree, it stores all discovered frequent items. FP-tree supports only prefix sharing while CFP – tree supports prefix sharing as well as suffix sharing. To compute and store all frequent items suffix sharing is feasible. CFP tree gives us an efficient access to frequent patterns.

ID	Itemsets	ID	Itemsets	ID	Itemsets
1	a:6	9	ac:3	17	acm:3
2	b:3	10	af:4	18	afm:3
3	c:3	11	am:5	19	afp:3
4	d:3	12	ap:3	20	amp:3
5	e:3	13	cm:3	21	fmp:3
6	f:5	14	fm:3	22	afmp:3
7	m:5	15	fp:4		
8	p:4	16	mp:3		

Table 2: Represents the Frequent patterns with Minimum Support of 3 (min_sup=3)

From the above two tables like table 1 and 2,we can clearly find out the data set with sample elements and also in the next table we can find out the Frequent patterns with a minimum support value of 3.After a deep analysis we have performed a CFP algorithm on the proposed data set so that we got a complete CFP tree structure ,which is clearly represented in the figure 3.

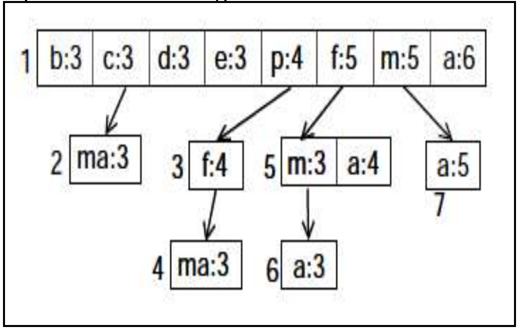


Figure 3 Represents the Proposed CFP – Tree Structure

Let us take a transactional dataset /database D with a minimum threshold value, this tree construction algorithm first scans the database D twice to construct a CFP – tree. In the very first scan, all the frequent patterns are counted and then stored in ascending frequency order. In the second scan, a conditional database is constructed for each frequent item.

Each entry in this tree must store several piece of information

1) The number of items in the entry
2) The support of the entry
3) Id for an entry by using pre ordinary
4) A pointer pointing to its child node

We may have both multiple entry nodes and singleton nodes. In a multiple entry node, each entry has only one item. A singleton node can have any number of items. The items, which are in singleton node, must have the same frequency means every entry in this tree represents one or more items / patterns with same support and every entry has a path from the root node. In the CFP – tree the supersets of a pattern should not appear in the right of the pattern. It should appear either on the left of the pattern or in the sub tree pointed by a pattern.

4. Proposed Representative Pattern Set Generation Algorithms

In this section we will mainly discuss about the proposed two algorithms which are used for generating frequent item sets as well as representative sets from the original data set. Now let us discuss about those two algorithms in detail as follows:

Generally in the process of data mining, users frequently scrutinize the individual patterns by hand. In those situations it is suitable to keep the number of representative patterns as elite as possible to reduce the human effort. The condition e-covered, allows a pattern to address its subsets only. If the removal of the condition $X1 \subset X2$, the algorithm further reduces the number of representative patterns and the human effort will be reduced. In other cases, if the subset constraint has been removed the support information will be lost but in the CFP tree, each node has its support information. It stores the support information along with items. Relaxing the condition $X1 \subset X2$, to further reduce the representative patterns, the modified condition of e- covered is

e*- covered:

1) Given two patterns X1 and X2, the distance between two patterns are defined as

$$D(X_1, X_2) = 1 - \frac{|T(X_1) \cap T(X_2)|}{|T(X_1) \cup T(X_2)|}$$

2) Given a real number **e** $\in$**[0,1)** and two patterns X1,X2. A pattern **X1** is e*-covers an another pattern **X2** if, **X1**$\subseteq$**X2** and **D(X1, X2)**$\leq$**e**.

Now we can discuss about the Minimum Representative Set algorithm in step by step process.

Algorithm 1: MINRPSet Algorithm

The proposed algorithm1 has following 6 steps, which as defined as follows:

1: Mine all patterns with support > minimum support;
2: Store them in a CFP-tree;
3: DFS_Search for all e*-covered patterns;

4: Remove all non-closed entries from e*-covered patterns;
5: Apply Greedy Set-cover algorithm on e*-covered patterns to find representative patterns;
6: Output representative Pattern Sets

The algorithm for retrieving the c(x)'s has been changed due to the relaxation of condition.

Proposed algorithm for generating C(X)'s:

This is defined clearly by using following steps:

Input:
Frequent Pattern Sets;
Output:
C(X)'s;
Description:

```
1. If a pattern X1 is frequent then

2. If X1.sup > all of its subsets AND
   X1.pattern superset of a pattern X2
   then

3. Mark X as closed pattern;

4. If E is marked as closed AND
   D(X1,X2)<= e then

5. Put Pattern X into C(X);
```

Now we can discuss about the second algorithm which is used in the proposed paper. This is as follows:

Algorithm 2: Greedy Set Cover Algo

This proposed Greedy Set Cover Algorithm is defined as follows in a step by step procedure.

Input:
C(X)'x
Output:
Representative pattern Sets
Description:

```
1: if a pattern X1 contains another pattern X2
   then

2: remove X2 from C(X);

3: Search_Greedy Algm(X1,X2);

4: until it covers all the C(X)'s;
```

From this algorithm we can clearly get an idea that initially we will take a set like C(X) as input and from that set we will retrieve the representative set as an output. For this we use the Search_Greedy Algo as the step 3 in order to filter out the item sets. This step 3 continues until all the representative item sets are retrived, once all the set is filtered out then it will be terminated.

5. Experimental Evaluation
Implementation is the stage where theoretical design is converted into programmatically way. Generally in the implementation stage we will divide the application into number of modules in order to make the application develop very easily. We have implemented the proposed concept on Java Platform in order to show the performance this proposed representative set generation algorithms for mining frequent items sets from a general data set.

Input for the Proposed Algorithms

The following two data sets are taking into account for evaluating the performance of our proposed algorithm compared with the primitive item set mining algorithms. Here we have took two data sets one is Chess.dat and other is Connect.dat

Dataset	Number of Transactions	Number of Items
Chess	3196	75
Connect	67557	129

After a deep analysis of the above two data sets on our proposed algorithms we finally came to an conclusion that our proposed algorithm gives better performance in terms of accuracy and time for generating the representative item sets over various existing algorithms. This is clearly represented by the below graph as follows:

From the below figure 4,we can clearly get an idea that the graph consists of two axis like X and Y-Axis ,where the X-axis is used to map various mining algorithms for mining frequent item sets from a data set and corresponding Y-Axis represent the time in milli seconds. Here in the X-Axis we have taken three algorithms in order to perform the operations on the proposed as well as existing algorithms. Of all the 3 algorithms we finally came to an conclusion that our proposed Representative Item Set generation algorithm takes less time for mining the frequent items from the data set compared with all other primitive algorithms. This is clearly shown in the below graph.

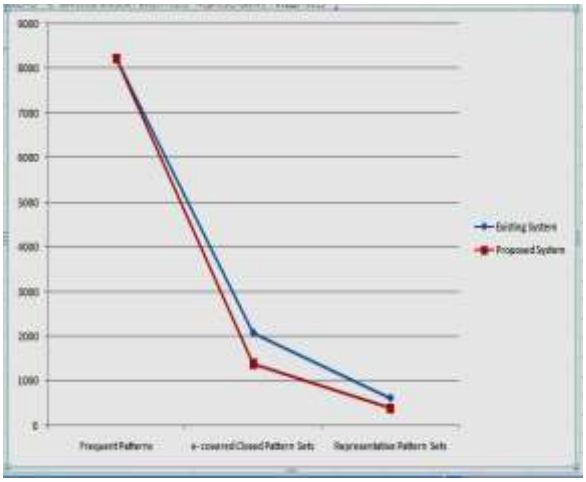


Figure 4.Represents the Performance of Representative pattern sets by using various algorithms

6. Conclusion

In this paper, we have described an algorithm called MINRPSet, for finding minimum representative pattern sets. This algorithm mines all the frequent patterns first, than it finds all the representative patterns in the post processing step, while RPLocal integrates the frequent pattern mining with representative pattern sets generation. Due to this MINRPSet produces minimum number of representative patterns than RPLocal. In the MINRPSet algorithm, it is easy to maintain all the patterns which are covered by representative patterns. This is useful to user to inspect individual patterns more efficiently. The efficiency of the algorithm is increased by modifying the e-covered condition.

7. References

[1] Guimei Liu, Haojun Zhang, Limsoon Wong, “Finding Minimum Representative Pattern Sets” KDD transaction on Finding representative pattern sets.
[2] Frequent itemset mining dataset repository. http://fimi.cs.helsinki.fi/data/.

[3] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal, "Discovering frequent closed itemsets for association rules," in *Proc. 7th ICDT*, Jerusalem, Israel, 1999, pp. 398–416.
[4] D. Xin, J. Han, X. Yan, and H. Cheng, "Mining compressed frequent-pattern sets," in *Proc. 31st Int. Conf. VLDB*, Trondheim, Norway, 2005, pp. 709–720.
[5] G. Liu, H. Lu, and J. X. Yu, "CFP-tree: A compact disk-based structure for storing and querying frequent itemsets," *Inf. Syst.*, vol. 32, no. 2, pp. 295–319, 2007.
[6] A. Bykowski and C. Rigotti, "A condensed representation to find frequent patterns," in *Proc. PODS*, New York, NY, USA, 2001, pp. 267-273.
[7] A. K. Poernomo and V. Gopalkrishnan, "CP-summary: A concise representation for browsing frequent itemsets," in *Proc. KDD*, New York, NY, USA, 2009, pp. 687–696.
[8] J. Wang, J. Han, and J. Pei, "Closet+: Searching for the best strategies for mining frequent closed itemsets," in *Proc. KDD*, New York, NY, USA, 2003, pp. 236–245.
[9] F. N. Afrati, A. Gionis, and H. Mannila, "Approximating a collection of frequent sets," in *Proc. KDD*, Washington, DC, USA, 2004, pp. 12–19.
[10] V. Chvatal, "A greedy heuristic for the set-covering problem," *Math. Oper. Res.*, vol. 4, no. 3, pp. 233–235, 1979.

8. About the Authors

Raju Sunkari is currently pursuing his 2 Years M.Tech in Computer Science and Engineering at Pydah College of Engineering and Technology, Gambheeram, Visakhapatnam. His area of interests includes Data Mining.

M.Kranthi Kumar is currently working as an Assistant Professor, in Computer Science and Engineering at Pydah College of Engineering and Technology, Gambheeram,Visakhapatnam. He has more than 3 years of experience in teaching field. His research interest includes Data Mining.

Dr.Ramesh Challagundla received his M.Sc. Degree in Phy.Electronics from Meerut University which is recognized as equivalent to B.E (ECE) in the year 1988, M.E. with specialization in Applied/Power Electronics, from Gulbarga University in1991, was granted A.M.I.E. In 1997 and Ph.D. from Andhra University College of Engineering, Andhra university, Visakhapatnam. He joined as service engineer in Hast Alloy Castings Ltd in the year 1990. After serving a year and half, he switched over to teaching and served as Lecturer in R.E.C. Affiliated to Gulbarga University during 1992-1993. He joined EEE Department, GITAM, Visakhapatnam and served as Lecturer during 1993-96. During 1996-97 he served as Lecturer in Bhilai Institute of Technology, during 1997-98 served as Assistant Professor in Birla Institute of Technology, Mesra, Ranchi, during 1998-2001 served as Assistant Professor in GITAM College of Engineering, Visakhapatnam and from 2001 onwards with ANITS.Currently working as Professor and Principal at Pydah College of Engineering and Technology, Gambheeram, Visakhapatnam. Ratified as Professor by the expert committee of Andhra University in the field of ECE constituted by Vice-Chancellor, who himself was the chairman for the selecton committee.