Item Set Mining using IMINE Index Support

*1Thadi Ananda Ravi Kumar, *2N Tulasi Raju , *3D.Chitti Babu , *4Subhakar Rao Golla

*1M.Tech Student, Dept of CSE, Swarndhra College of Engg & Tech, Seetharampuram, W.G..Dt (A.P).India.
*2Assistant Professor, Dept of CSE, Swarndhra College of Engg & Tech, Seetharampuram, W.G..Dt(A.P).India.
*3 Assistant Professor, Dept of IT, Swarndhra College of Engg & Tech, Seetharampuram, W.G..Dt(A.P).India.
*4 Assistant Professor, Dept of IT, Swarndhra College of Engg & Tech, Seetharampuram, W.G..Dt (A.P).India.

Abstract- We are going to propose tight integration of item set extraction in a relational open source DBMS, by exploiting its physical level access method which is called as IMINE Index, Using which we can represent the original database Since no constraint is enforced during the index creation phase. To reduce the I/O cost, data accessed together during the same extraction phase are clustered on the same disk block. The IMINE index structure can be efficiently exploited by different item set extraction algorithms. Presently FP-growth and LCM v.2 are two algorithms which are particularly supported by IMINE data access methods.

Keywords- FP-Growth, IMINE, Datasets, Tree, Association rule.

I. INTRODUCTION

With the wide use of computers, scanners and data base technique, human accumulated a great deal of historical data. These data look simple at the surface of them, but, there is much valuable information behind them. In data prediction, business decision and resource management, the knowledge and rule behind these data are very useful. But, if we still use traditional methods of statistical and analyses, these useful information can't be discovered or can be found in infinite time. Hence data mining has been proposed on this occasion. As one of the main research patterns in the field of data mining, association rules are used to determine the relationships of a set of item, to find out valuable information. Frequent item mining, the main task of the association rule mining, the efficiency of which is the difficult problem. In this paper, relevant knowledge of frequent itemset mining is introduced and some classic algorithms are analyzed in detail. For the maximum frequent contains all the frequent item sets, this paper focuses on how to mining maximum frequent item sets, the maximum frequent mining from generating FP-tree, the prune strategy, superset checking, first searching strategy, reducing dimension are deeped researched.

II. PREVIOUS WORK

Association rule mining discovers correlations among data items in a transactional database D. Each transaction in D is a set of data items. Association rules are usually represented in the form A! B, where A & B are item sets, i.e., sets of data items. Item sets are characterized by their frequency of occurrence in D, which is called support. Research activity usually focuses on defining efficient algorithms for item set extraction, which represents the most computationally intensive knowledge extraction task in association rule mining. The data to be analyzed is usually stored into binary files, possibly extracted from a DBMS. Most algorithms exploit ad hoc main memory data structures to efficiently extract item sets from a flat file. Recently, disk-based extraction algorithms have been proposed to support the extraction from large data sets but still dealing with data stored in flat files. To reduce the computational cost of item set extraction, different constraints may be enforced among which the most simple is the support constraint, which enforces a threshold on the minimum support of the extracted item sets. Relational DBMSs exploit indices, which are ad hoc data structures, to enhance query performance and support the execution of complex queries. In this paper, we propose a similar approach to support data mining queries. The IMINE index (Item set-Mine index) is a novel data structure that provides a compact and complete representation of transactional data supporting efficient item set extraction from a relational DBMS. It is characterized by the following properties:

- It is a covering index. No constraint (e.g., support constraint) is enforced during the index creation phase. Hence, the extraction can be performed by means of the index alone, without accessing the original database. The data representation is complete and allows reusing the index for mining item sets with any support threshold.
- The IMine index is a general structure which can be efficiently exploited by various item set extraction algorithms. These algorithms can be characterized by different in-memory data representations (e.g., array list, prefix-tree) and

techniques for visiting the search space. Data access functions have been devised for efficiently loading in memory the index data. Once in memory, data is available for item set extraction by means of the algorithm of choice. We implemented and experimentally evaluated the integration of the IMine index in FP-growth and LCM v.2. Furthermore, the IMine index also supports the enforcement of various constraint categories.

- The IMine physical organization supports efficient data access during item set extraction. Correlation analysis allows us to discover data accessed together during pattern extraction. To minimize the number of physical data blocks read during the mining process, correlated information is stored in the same block.
- IMine supports item set extraction in large data sets.

III. PROPOSED WORK

In this section we are going to propose the techniques which we are going to use in the system and we consider example data set as in Fig.1 to illustrate how the techniques can be implemented .And the techniques are as follows:

a) Frequent Item Set Extraction.
b) I Tree Module.
c) I BTree Module.

TID	ItemsID
1	g,b,h,e,p,v,d
2	e,m,h,n,d,b
3	p,e,c,i,f,o,h
4	j,h,k,a,w,e
5	n,b,d,e,h
6	s,a,n,r,b,u,i
7	b,g,h,d,e,p
8	a,i,b
9	f,i,e,p,c,h
10	t,h,a,e,b,r
11	a,r,e,b,h
12	z,b,i,a,n,r
13	b,e,d,p,h

Fig.1: Example Data Set

Frequent Item Set Extraction:

This section describes how frequent item set extraction takes place on the IMine index. We present two approaches, denoted as FP-based and LCM-based algorithms, which are an adaptation of the FP-Growth algorithm and LCM v.2 algorithm, respectively.

➢ ***FP-based algorithm***

The FP-growth algorithm stores the data in a prefix-tree structure called FP-tree. First, it computes item support. Then, for each transaction, it stores in the FP-tree its subset including frequent items. Items are considered one by one. For each item, extraction takes place on the frequent-item projected database, which is generated from the original FP-tree and represented in a FP-tree based structure.

➢ ***LCM-based algorithm***

The LCM v.2 algorithm loads in memory the support-based projection of the original database. First, it reads the transactions to count item support. Then, for each transaction, it loads the subset including frequent items. Data are represented in memory by means of an array-based data structure, on which the extraction takes place.

I Tree Module

The Item set-Tree (I-Tree) is a prefix-tree which represents relation R by means of a succinct and lossless compact structure. Implementation of the I-Tree is based on the FP-tree data structure, which is very effective in providing a compact and lossless representation of relation R. However, since the two index components are designed to be independent, alternative I-Tree data structures can be easily integrated in the IMine index as shown in Fig.2

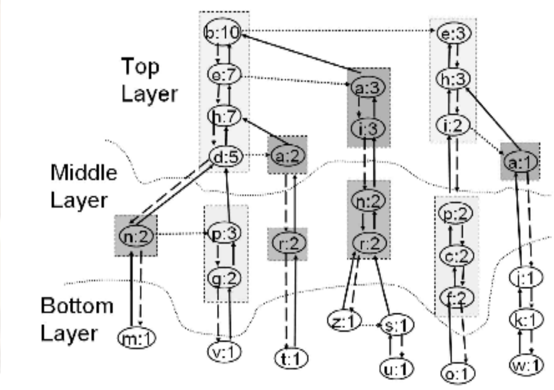

Fig.2: IMINE Index for Example data set of I-Tree

The I-Tree associated to relation R is actually a forest of prefix-trees, where each tree represents a group of transactions all sharing one or more items. Each node in the I-Tree corresponds to an item in R. Each path in the I-Tree is an ordered sequence of nodes and represents one or more transactions in R. Each item in relation R is associated to one or more I-Tree nodes and each transaction in R is represented by a unique I-Tree path.

I BTree Module

The Item-Btree (I-Btree) is a B+Tree structure which allows reading selected I-Tree portions during the extraction task. For each item, it stores the physical locations of all item occurrences in the I-Tree. Thus, it supports efficiently loading from the I-Tree the transactions in R including the item. I-Btree allows selectively accessing the I-Tree disk blocks during the extraction process as in the Fig.3. It is based on a B+Tree structure. For each item i in relation R, there is one entry in the I-Btree.

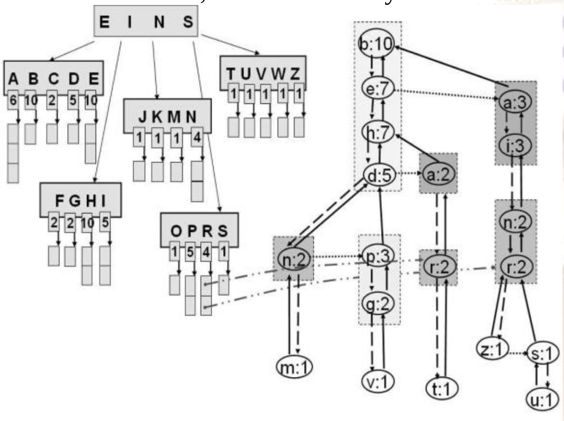

Fig.3: IMINE Index for Example data set of I-BTree

IV. DESIGN & IMPLEMENTATION OF THE SYSTEM

Design is a meaningful engineering representation of something that is to be built. Design creates a representation or model, provides detail about software data structure, architecture, interfaces and components that are necessary to implement a system. The use case model in the Fig.4 defines the outside (actors) and inside (use case) of the system's behavior.

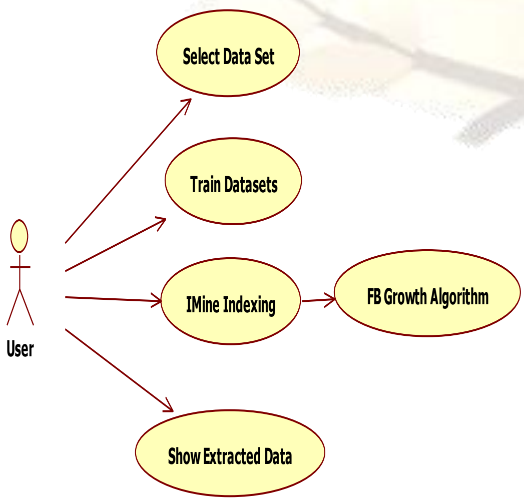

Fig.4: The Interoperabailty Usecase Diagram of the system

Activity diagram can also be used to represent a class's method implementation. A token represents an operation. An activity is shown as a round box containing the name of the operation. An outgoing solid arrow attached to the end of activity symbol indicates a transition triggered by the completion as shown in Fig.5.

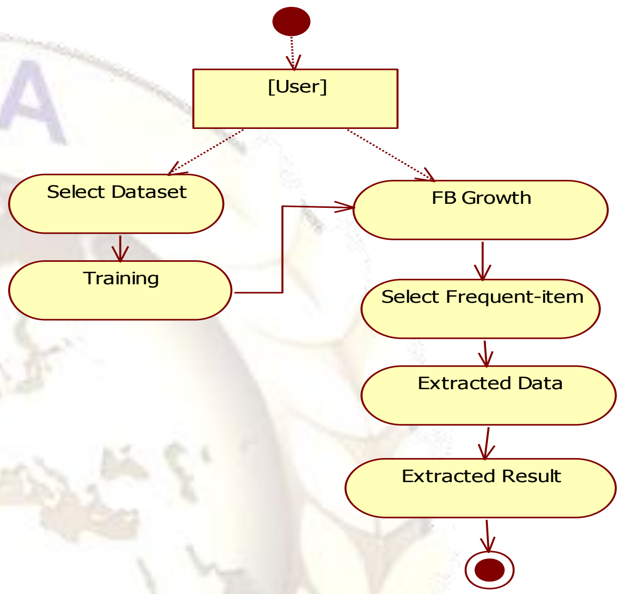

Fig.5: The Interoperabailty Usecase Diagram of the system

In this coming section we are going to discuss about the implementation of how the Mining has been done in-order to identify the Association rule text information is hidden. The below given code in fig.6 will generate the Fp Tree implementation.

```
public FPtree(String[] args)
{
super(args);
rootNode = new FPtreeNode();
headerTable = new FPgrowthHeaderTable
[numOneItemSets + 1];
for (int index = 1; index < headerTable.length; index++)
{
headerTable[index] = new FPgrowthHeaderTable((short)
index);
}          }
public void createFPtree()
{
headerTable = new FPgrowthHeaderTable
[numOneItemSets + 1];
for (int index = 1; index < headerTable.length; index++)
{
headerTable[index] = new FPgrowthHeaderTable((short)
index);
}
for (int index = 0; index < dataArray.length; index++)
{
if (dataArray[index] != null)
addToFPtree(rootNode, 0, dataArray[index], 1,
headerTable);
}          }
```

Fig.6: Implementation of the Fp Tree

The below given code in Fig.7 is used to implement Associate rule mining of the system.

```
public AssocRuleMining(String[] args) {
for (int index = 0; index < args.length; index++)
idArgument(args[index]);
if (errorFlag)
CheckInputArguments();
else
outputMenu();
}
protected void idArgument(String argument) {
if (argument.charAt(0) == '-') {
char flag = argument.charAt(1);
argument = argument.substring(2, argument.length());
switch (flag) {
case 'C':
confidence = Double.parseDouble(argument);
break;
case 'F':
fileName = argument;
break;
case 'S':
support = Double.parseDouble(argument);
break;
default:
System.out.println("INPUT ERROR: Unrecognise command "
+ "line argument -" + flag + argument);
errorFlag = false;
}                        } else {
System.out.println("INPUT ERROR: All command line arguments "
+ "must commence with a '-' character (" + argument + ")");
errorFlag = false;          }          }
```

Fig.7: Implementation of Associate Rule Mining

V. RESULTS

The following are the obtained results from the proposed system.

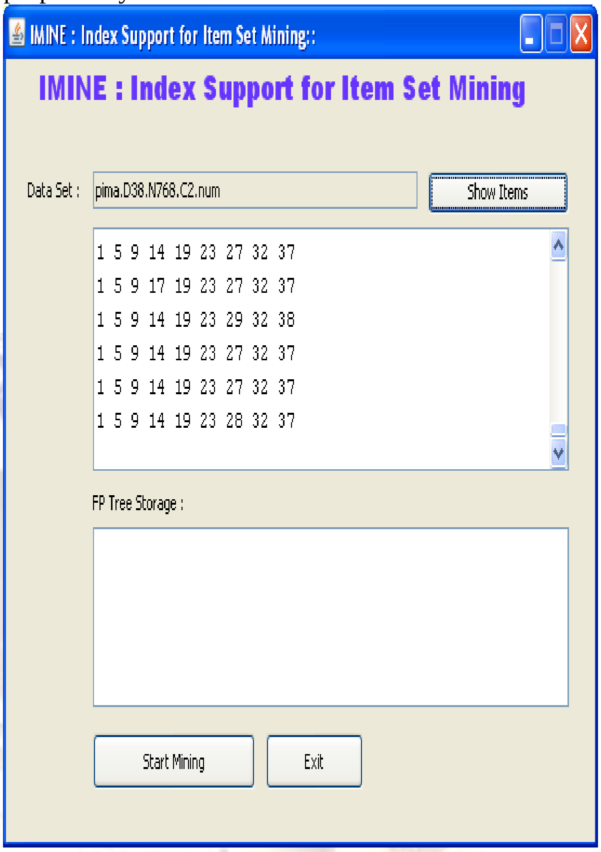

Fig.8: Identifying the Data Sets

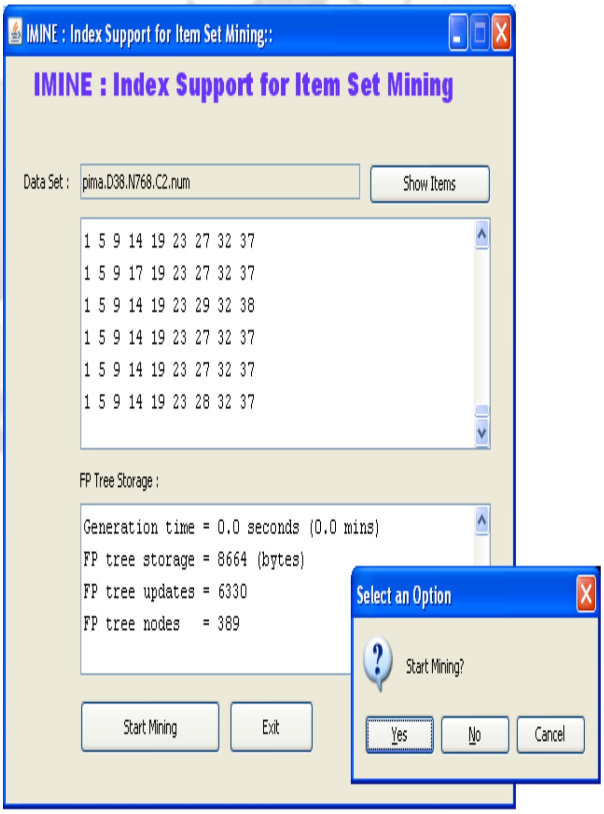

Fig.9: Calculating FP Tree Storage and Starting

Mining

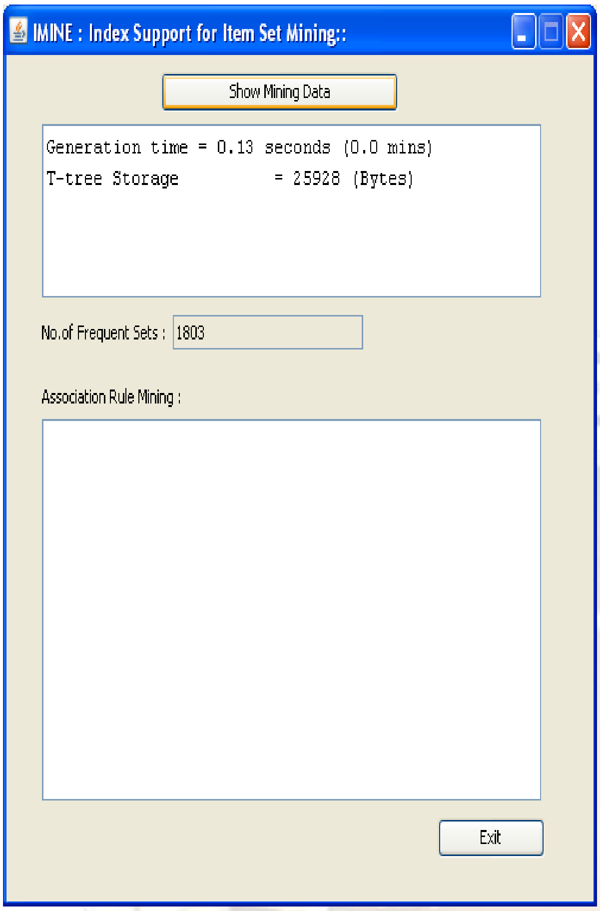

Fig.10: Display of Mining Data and Number of Frequent Sets

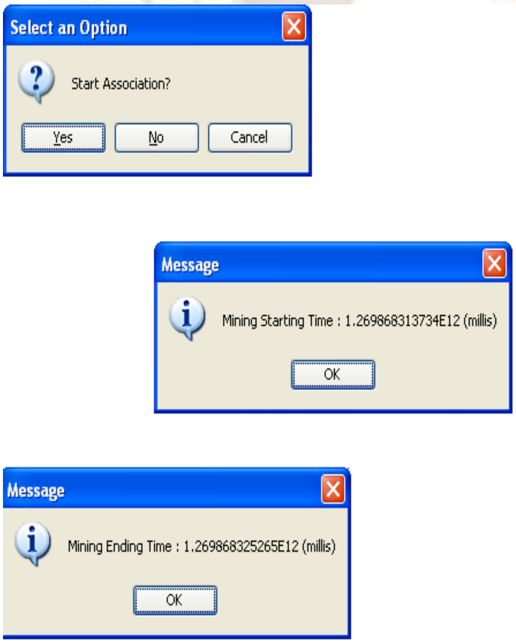

Fig.11: Starting Association and display of mining starting & ending time

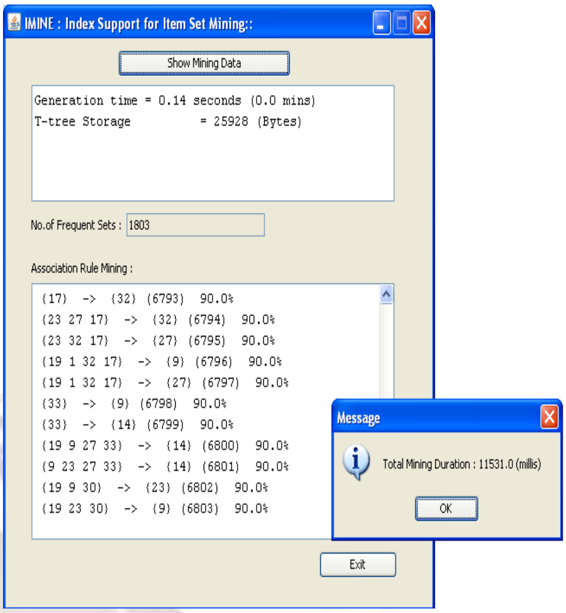

Fig.12: Calculation of Association Rule Mining and Total Mining Duration

VI. CONCLUSION

The IMine index is a novel index structure that supports efficient item set mining into a relational DBMS. It has been implemented into the open source DBMS, by exploiting its physical level access methods. The IMine index provides a complete and compact representation of transactional data. It is a general structure that efficiently supports different algorithmic approaches to item set extraction. Selective access of the physical index blocks significantly reduces the I/O costs and efficiently exploits DBMS buffer management strategies. This approach, albeit implemented into a relational DBMS, yields better performance than the state-of-the-art algorithms accessing data on a flat file and is characterized by a linear scalability also for large data sets.

REFERENCES

[1] H. Toivonen, “Sampling Large Databases for Association Rules,” Proc. 22nd Int’l Conf. Very Large Data Bases (VLDB ’96), pp. 134-145, 1996.
[2] M. El-Hajj and O.R. Zaiane, “Inverted Matrix: Efficient Discovery of Frequent Items in Large Datasets in the Context of Interactive Mining,” Proc. Ninth ACM SIGKDD Int’l Conf. Knowledge Discovery and Data Mining (SIGKDD), 2003.
[3] G. Grahne and J. Zhu, “Mining Frequent Itemsets from Secondary Memory,” Proc. IEEE Int’l Conf. Data Mining (ICDM ’04), pp. 91-98, 2004.
[4] G. Ramesh, W. Maniatty, and M. Zaki, “Indexing and Data Access Methods for Database Mining,” Proc. ACM SIGMOD Workshop Data Mining and Knowledge Discovery (DMKD), 2002.

[5] Y.-L. Cheung, "Mining Frequent Itemsets without Support Threshold: With and without Item Constraints," IEEE Trans. Knowledge and Data Eng., vol. 16, no. 9, pp. 1052-1069, Sept. 2004.
[6] G. Cong and B. Liu, "Speed-Up Iterative Frequent Itemset Mining with Constraint Changes," Proc. IEEE Int'l Conf. Data Mining (ICDM '02), pp. 107-114, 2002.
[7] C.K.-S. Leung, L.V.S. Lakshmanan, and R.T. Ng, "Exploiting Succinct Constraints Using FP-Trees," SIGKDD Explorations Newsletter, vol. 4, no. 1, pp. 40-49, 2002.
[8] R. Srikant, Q. Vu, and R. Agrawal, "Mining Association Rules with Item Constraints," Proc. Third Int'l Conf. Knowledge Discovery and Data Mining (KDD '97), pp. 67-73, 1997.
[9] S. Sarawagi, S. Thomas, and R. Agrawal, "Integrating Mining with Relational Database Systems: Alternatives and Implications," Proc. ACM SIGMOD, 1998.
[10] J. Han, Y. Fu, W. Wang, K. Koperski, and O. Zaiane, "DMQL: A Data Mining Query Language for Relational Databases," Proc. ACM SIGMOD Workshop Data Mining and Knowledge Discovery (DMKD), 1996.
[11] R. Meo, G. Psaila, and S. Ceri, "A New SQL-Like Operator for Mining Association Rules," Proc. 22nd Int'l Conf. Very Large Data Bases (VLDB), 1996.

AUTHOR DETAILS

Ananda Ravi Kumar Thadi received the master's degree in computer applications from andhra university, pursuing m.tech (computer science and engineering). he is an P.G STUDENT, DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, swarandhra collage of engineering & technology, seetharampuram, narasapuram,ap. india. his research areas of interests are data warehousing and data mining

Tulasi Raju Nethala received the master's degree in computer science and engineering from jntu kakinada. he is an asst. professor in the department of computer science in swarnandhra college of engineering & technology. his research areas of interests are in data mining .

Chittibabu Dondapati received the mtech degree in information technology from satyabhama university, chennai. he is currently working as asst.prof in DEPARTMENT OF INFORMATION TECHNOLOGY, swarandhra collage of engineering & technology, seetharampuram, narasapuram,ap. india. his research areas of interests are data warehousing and data mining

Subhakar Rao Golla received the m.tech degree in computer science and engineering from jntu kakinada. he is currently working as asst.prof in DEPARTMENT OF INFORMATION TECHNOLOGY, swarandhra collage of engineering & technology, seetharampuram, narasapuram,ap. india. his research areas of interests are data warehousing and data mining.