Vol. 5, No. 5, May 2014 ISSN 2079-8407
Journal of Emerging Trends in Computing and Information Sciences

http://www.cisjournal.org

A Survey of Distributed Association Rule Mining Algorithms

[1] Vinaya Sawant, [2] Ketan Shah
[1]Asstt Prof., Department of Information Technology, DJSCE, Mumbai University
[2] Assoc. Prof., Department of Information Technology, MPSTME, NMIMS University

[1] vinaya.sawant@djsce.ac.in, [2] KetanShah@nmims.edu

ABSTRACT

Association Rule Mining is a popular and well researched method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using different measures of interestingness. Most ARM algorithms focus on a sequential or centralized environment where no external communication is required. Distributed ARM algorithms, aim to generate rules from different data sets spread over various geographical sites; hence, they require external communications throughout the entire process. Distributed ARM is one of the major research fields of Data Mining (DM). DARM algorithm efficiency is highly dependent on data distribution. The paper reviews different algorithms developed for DARM and also discusses the different ways in which data is distributed. Agents are software entities developed to make distributed computing more efficient. They have also been used in Data Mining. The paper discusses the role of agents in DARM.

Keywords: *Association Rule Mining, Distributed Association Rule Mining, Agents in Data Mining.*

1. INTRODUCTION

Though information technology (IT) is considered one of the greatest blessings of technology at current era, rapid increase in information in various formats and at different locations may explode the whole arena of IT if it is not supervised properly. Data mining is one of the means to utilize information by discovering underlying hidden useful knowledge from information. Among different techniques proposed in Data Mining, association rule mining (ARM) is one of the popular techniques for mining data.

Association rule mining, one of the most important and well researched techniques of data mining. It aims to extract interesting correlations, frequent patterns, associations or casual structures among sets of items in the transaction databases or other data repositories. Association rules are widely used in various areas such as telecommunication networks, market and risk management, inventory control etc.

Association rule mining problem is usually decomposed into two sub problems. One is to find those item sets whose occurrences exceed a predefined threshold in the database based on the given support; those item sets are called frequent or large item sets. The second problem is to generate association rules from those large item sets with the constraints of minimal confidence. Rules generated should satisfy the minimum confidence [1].

2. DISTRIBUTED DATA MINING

Launched Data mining algorithms deal predominantly with simple data formats (typically flat files); there is an increasing amount of focus on mining complex and advanced data types such as object-oriented, spatial, web and temporal data. Another aspect of this growth and evolution of data mining systems is the move from stand-alone systems using centralized and local computational resources towards supporting increasing levels of distribution. As data mining technology matures and moves from a theoretical domain to the practitioner's arena there is an emerging realization that distribution is very much a factor that needs to be accounted for.

Databases in today's information age are inherently distributed. Organizations that operate in global markets need to perform data mining on distributed data sources (homogeneous / heterogeneous) and require cohesive and integrated knowledge from this data. Such organizational environments are characterized by a geographical separation of users from the data sources. This inherent distribution of data sources and large volumes of data involved inevitably leads to exorbitant communications costs. Therefore, it is evident that traditional data mining model involving the co-location of users, data and computational resources is inadequate when dealing with distributed environments. The development of data mining along this dimension has lead to the emergence of distributed data mining.

The need to address specific issues associated with the application of data mining in distributed computing environments is the primary objective of distributed data mining. Broadly, data mining environments consist of users, data, hardware and the mining software (this includes both the mining algorithms and any other associated programs). Distributed data mining addresses the impact of distribution of users, software and computational resources on the data mining process. There is general consensus that distributed data mining is the process of mining data that has been partitioned into one or more physically/geographically distributed subsets.

The significant factors, which have led to the emergence of distributed data mining from centralized mining, are as follows:

- The need to mine distributed subsets of data, the integration of which is non- trivial and expensive.
- The performance and scalability bottle necks of data mining.
- Distributed data mining provides a framework for scalability, which allows the splitting up of larger datasets with high dimensionality into smaller subsets that require computational resources individually.

Distributed Data Mining (DDM) [2] is a branch of the field of data mining that offers a framework to mine distributed data paying careful attention to the distributed data and computing resources. In the DDM literature, one of two assumptions is commonly adopted as to how data is distributed across sites: homogeneously and heterogeneously. Both viewpoints adopt the conceptual viewpoint that the data tables at each site are partitions of a single global table. In the homogeneous case, the global table is horizontally partitioned. The tables at each site are subsets of the global table; they have exactly the same attributes. In the heterogeneous case the table is vertically partitioned, each site contains a collection of columns (sites do not have the same attributes). However, each tuple at each site is assumed to contain a unique identifier to facilitate matching. It is important to stress that the global table viewpoint is strictly conceptual. It is not necessarily assumed that such a table was physically realized and partitioned to form the tables at each site.

3. DISTRIBUTED ASSOCIATION RULE MINING

Modern organizations are geographically distributed. Typically, each site locally stores its ever increasing amount of day-to-day data. Using centralized data mining to discover useful patterns in such organizations' data isn't always feasible because merging data sets from different sites into a centralized site incurs huge network communication costs. Data from these organizations are not only distributed over various locations but also vertically fragmented, making it difficult if not impossible to combine them in a central location. Distributed data mining has thus emerged as an active subarea of data mining research.

Distributed ARM algorithms aim to generate rules from different data sets spread over various geographical sites; hence, they require external communications throughout the entire process. DARM algorithms must reduce communication costs so that generating global association rules costs less than combining the participating sites' data sets into a centralized site.

However, most DARM algorithms don't have an efficient message optimization technique, so they exchange numerous messages during the mining process.

Following paragraph describes various Distributed Association Rule Algorithms used in research work along with the nature of datasets used in the algorithms.

3.1 Count Distribution (CD) Algorithm

CD algorithm can be summarized into five major stages:

a. Each processor generates candidate itemset Ck based on globally frequent large itemset Lk-1.
b. Each processor computes local support count for Ck by passing through the transactions in the database.
c. All processors exchange their Ck counts to develop global Ck.
d. Each processor computes Lk from Ck.
e. Each processor takes the decision either to continue or to stop. Decision will be the same since they have identical Lk.

The Count Distribution (CD) algorithm is a simple data-parallelism algorithm. It uses the sequential Apriori algorithm in a parallel environment and assumes data sets are horizontally partitioned among different sites.

The CD algorithm's main advantage is that it doesn't exchange data tuples between processors it only exchanges the counts. In the first scan, each processor generates its local candidate itemset depending on the items present in its local partition. The algorithm obtains global counts by exchanging local counts with all other processors **[1]**.

3.2 Fast Distribution Mining Algorithm (FDM)

The main idea of this algorithm can be summarized as follows:

a. Computing candidate set: Each site generates candidate set based on globally large (k-1)-item sets and locally large (k-1)-item sets using Apriori algorithm.
b. Local pruning: For each item in the candidate set: if the support of the itemset is larger than minimum support, that particular item is added in the locally large k-item sets.
c. Count exchange: Each site broadcasts locally frequent large item sets to all other sites.
d. Globally frequent large itemset computation: Each site computes globally large k-item sets which is utilized for the following iteration.

The commonly used datasets for the FDM algorithm is the horizontally partitioned data on different sites. The performance of FDM over CD with respect to communication cost and time is better. Also, the experiments also show that the result of the algorithm varies with respect to the number of transactions and the number of processors in the distributed environment.

FDM's main advantage over CD is that it reduces the communication overhead FDM generates fewer candidate item sets compared to CD, when the number of disjoint candidate item sets among various sites is large. However, we can only achieve this when different sites have no homogeneous data sets.

FDM's message optimization techniques require some functions to determine the polling site, which could cause extra computational cost when each site has numerous local frequent item sets. Furthermore, each polling site must send a request to remote sites other than the originator site to find an item set's global support counts, increasing message size when numerous remote sites exist **[1]**.

3.3 Optimized Distributed Association Rule Mining

The algorithm can be summarized as below:

a. ODAM first computes support counts of 1-itemsets from each site in the same manner as it does for the sequential Apriori.
b. It then broadcasts those item sets to other sites and discovers the global frequent 1-itemsets.
c. Subsequently, each site generates candidate 2-itemsets and computes their support counts.
d. At the same time, ODAM also eliminates all globally infrequent 1-itemsets from every transaction and inserts the new transaction (that is, a transaction without infrequent 1-itemset) into memory.
e. While inserting the new transaction, it checks whether that transaction is already in the memory. If it is, ODAM increases that transaction's counter by one.
f. Otherwise, it inserts the transaction with a count equal to one into the main memory. After generating support counts of candidate 2-itemsets at each site, ODAM generates the globally frequent 2-itemsets.
g. It then iterates through the main memory (transactions without infrequent 1-itemsets) and generates the support counts of candidate item sets of respective length. 8.
h. Next, it generates the globally frequent item sets of that respective length by broadcasting the support counts of candidate item sets after every pass.

The existing research work done on this algorithm considered the horizontally partitioned datasets. Unlike other algorithms, ODAM offers better performance by minimizing candidate itemset generation costs. It achieves this by focusing on two major DARM issues communication and synchronization.

Communication is one of the most important DARM objectives. DARM algorithms will perform better if there is a reduction in communication (for example, message exchange size) costs. Synchronization forces each participating site to wait a certain period until globally frequent itemset generation completes **[3]**.

3.4 ODAM for XML Data

In XML data, multiple nesting is a problem that needs to be handled properly. Consider a file of sales receipts from a grocery chain. The grocery chain may want to group by the following information: Date, Store Id, Register, and Individual Sale. Any permutation of these attributes would be a logical construction of a file in XML. With the individual sale as the attribute of most interest, consider the various nesting depths at which it may be located. At any node in an XML 'tree' the sub tree can be viewed as a record, relative to other records at that depth, or other records with similar record tags. This provides assurance that mining will be done on the correct nesting depth (along with other nesting depths also).

However there is a potential for redundancy. It becomes more evident in highly nested files. In a highly nested XML file, the same set of leaf nodes may be involved in as many different records as there are nestings. The below Algorithm is used to derive General Association Rules from XML Data.

a. First the record ids are assigned per record type.
b. A basketSet is constructed for each type of record encountered.
c. An empty record Type List and an empty RID List is taken first to start. These lists are parallel, in that the record Type at position n of the record Type List is associated with the RID at position n of the RID List.
d. A single path from root to leaf is considered. As the algorithm progresses along this path, it examines each node.
e. If the node is not a leaf, it looks at the node type (record Type) and asks the basketSet associated with this record Type for a new RID.
f. It then adds the record Type to the end of the record Type list and the RID to the end of the RID List.
g. If the node is a leaf (consider a leaf to be of the form <purchase>pen</purchase>) loop through the RID List and record Type list to build Baskets.

The number of messages that ODAM exchanges among various sites to generate the globally frequent item sets in a distributed environment, partition the original data set into n partitions. To reduce the dependency among different partitions, each one contains only some percent of the original data set's transactions. So, the number of identical transactions among different partitions is very low **[5]**.

3.5 AprTidRec algorithm

AprTidRec proposed is similar to Apriori, the difference between them is that Apriori includes join step and pruning step while AprTidRec include only join step when generate frequent itemset. In AprTidRec, a record

structure called tidRec is defined for each candidate frequent itemset. The tidRec of itemset I consist of TID of the transactions who contain itemset I. I.tidRec is the tidRec of itemset I. The tidRec of 1-itemset can be got by scanning the transaction database. The structure of the record in the algorithm is < I, tidRec, count >, I.count is the support of the itemset I, it is equal to the length of tidRec that is count = | tidRec |. When generating candidate frequent k-itemset from frequent k-1-itemset, the tidRec and support of the candidate frequent k-itemset can be derived from the intersection of the tidRec of the two k-1-itemsets.

AprTidRec-algorithm description: (Ck is candidate frequent k-itemset, Lk is frequent k itemset)

i. K=1, Lk= ϕ
ii. for all item sets I1€ Lk-1 do begin
iii. for all item sets I2€ Lk-1 do begin
iv. If I1.iteml=I2.iteml ^ I1.item2=I2.item2 ^ … ^ I1.item k-2=I2.item k-2 ^ I1.item k-1 < I2 .itemk-1; then
vi. begin
vii. Ck.itemsets= I1.item1.I1.item2…I1.itemk-1.I2.itemk-1
viii. Ck.tidRec = I1.tidRec ∩ I2.tidRec
ix. Ck.count= | Ck.tidRec |
x. end
xi. if(Ck.count ≥| D | * minsup)then
xii. Lk= Lk { Ck }
xiii. end
xiv. end

From the above algorithm, for generating global frequent k-itemset, scan the local databases only once (during constructing the new storage structure) and prune step is not required. So I/O spending is saved, and time complexity of the algorithm is reduced and efficiency is improved. But reduction of time complexity is at the cost of increase of space complexity. Each candidate itemset need a tidRec structure in the algorithm so large of memory space is required if transaction database is huge **[4]**.

In one of the experiments on this algorithm, horizontally partitioned junk mail database was considered.

The following table lists the DARM algorithms with its advantages and limitations:

Algorithm	Data Distribution	Advantages	Limitations
CD	Horizontally Partitioned Data	It doesn't exchange data tuples between processors it only exchanges the counts	Generates higher number of candidate sets and larger amount of communication overhead. It does not use the memory of the system effectively.
FDM	Horizontally Partitioned Data	It reduces the communication overhead. It generates fewer candidate item sets compared to CD.	Extra computational cost required in message passing when each site has numerous local frequent item sets. Message size increases when numerous remote sites exist.
ODAM	Horizontally Partitioned Data	ODAM offers better performance by minimizing candidate itemset generation costs. It achieves this by focusing on two major DARM issues communication and synchronization	Privacy and Security issues are not considered
ODAM for XML Data	Horizontally Partitioned Data	It provides an efficient method for generating association rules from different datasets, distributed among various sites. The Response time	Privacy and Security issues are not considered

		of the parallel and distributed data mining task on XML data is carried out by the time taken for communication, computation cost involved. An improved response time is achieved for the taken XML data.	
AprTidRec	Horizontally Partitioned Data	I/O spending is saved, and time complexity of the algorithm is reduced and efficiency is improved.	There is a reduction of time complexity but at the cost of increase of space complexity. Large memory space is required if transaction database is huge.

4. DATA DISTRIBUTION IN DARM

There are several ways in which data distribution can occur, and these require different approaches to model construction, includes horizontal and vertical partitioning

4.1 Horizontal Data Distribution

The most straight forward form of distribution is horizontal partitioning, in which different records are collected at different sites, but each record contains all of the attributes for the object it describes. This is the most common and natural way in which data may be distributed. For example, a multinational company deals with customers in several countries, collecting data about different customers in each country. It may want to understand its customers worldwide in order to construct a global advertising campaign **[6]**.

4.2 Vertical Data Distribution

The second form of distribution is vertical partitioning, in which different attributes of the same set of records are collected at different sites. Each site collects the values of one or more attributes for each record and so, in a sense, each site has a different view of the data. For example, a credit-card company may collect data about transactions by the same customer in different countries and may want to treat the transactions in different countries as different aspects of the customers total card usage. Vertically partitioned data is still rare, but it is becoming more common and important.

The data set is partitioned into horizontal or vertical partitions that can be distributed among a number of processors and independently processed, to identify local item sets, on each process **[6]**.

4.3 Hybrid Data Distribution

In most cases simple horizontal or vertical fragmentation of a DB schema will not be sufficient to satisfy the requirements of the applications. Mixed fragmentation (hybrid fragmentation) consists of a horizontal fragment followed by a vertical fragmentation, or a vertical fragmentation followed by a horizontal fragmentation.

If the distributed data cannot be horizontally fragmented because there is no guarantee that every site will include the same set of items, and if different distributed sites also refer to the same object multiple times (e.g., investigative reports about different crimes committed by the same individual). On the other hand, the data is not vertically fragmented either, because there is no one-to-one mapping connecting records in the distributed databases. In addition, the (local) 'schema' for each individual document varies, and no clean division of all objects' items into identical sets can be made as required for vertically fragmented data. As a result, the data is neither vertically nor horizontally fragmented, but is present in a form we term a hybrid fragmentation **[7]**.

4.4 Multi Dimensional Inter Transactional

Intra Transaction associations are the associations among items within the same transaction, where the notion of the transaction could be the items bought by the same customer, the events happened on the same day, and so, on. However, inter-transaction association describes the association relationships among different transactions, such as "If company A's stock goes up on day 1, B's Stock will go down on day 2, but go up on day by 4". Here, we treat associated items belongs to different transactions. Such an inter-transaction association can be extended to associate multiple contextual properties in the same rule, so that multi-dimensional inter-transaction associations can be defined or discovered. For example, "After McDonald and Burger King open branched, KFC will open a branch two months later and one mile away", which involves two dimensions: time and space. Mining inter-transactions poses more challenges on efficient processing than mining intra-transaction associations **[8]**.

5. SIGNIFICANCE OF INTELLIGENT AGENTS IN DATA MINING

Agents are defined as software or hardware entities that perform some set of tasks on behalf of users with some degree of autonomy. In order to work for somebody as an assistant, an agent has to include a certain amount of intelligence, which is the ability to choose among various courses of action, plan, communicate, adapt to changes in the environment, and learn from experience. In general, an intelligent agent can be described as consisting of a sensing element that can receive events, a recognizer or classifier that determines which event occurred, a set of logic ranging from hard-coded programs to rule-based inferencing, and a mechanism for taking action.

Data mining agents seek data and information based on the profile of the user and the instructions she gives. A group of flexible data-mining agents can co-operate to discover knowledge from distributed sources. They are responsible for accessing data and extracting higher-level useful information from the data. A data mining agent specializes in performing some activity in the domain of interest. Agents can work in parallel and share the information they have gathered so far.

Pericles A. Mitkas et al's [9] work on Software agent technology has matured enough to produce intelligent agents, which can be used for controlling a large number of concurrent engineering tasks. Multi-agent systems are communities of agents that exchange information and data in the form of messages.

The agents' intelligence can range from rudimentary sensor monitoring and data reporting, to more advanced forms of decision making and autonomous behavior. The behavior and intelligence of each agent in the community can be obtained by performing data mining on available application data and the respected knowledge domain. An Agent Academy a software platform is designed for the creation, and deployment of multiagent systems, which combines the power of knowledge discovery algorithms with the versatility of agents. Using this platform, agents are equipped with a data-driven inference engine, can be dynamically and continuously trained. Three prototype multi-agent systems are developed with Agent Academy.

Agent-based systems belong to the most vibrant and important areas of research and development to have emerged in information technology. Because of the lively extensive spreading of directions in research no publicly accepted solid definitions of agent-based systems and their elements – agents is provided.

Intelligent Agent (IA) refers to a software agent that exhibits some form of artificial intelligence. According to Wooldridge intelligent agents are defined as agents, capable of flexible autonomous action to meet their design objectives. They must involve:

- Reactivity: to perceive and respond in a timely fashion to changes occurring in their environment in order to satisfy their design objectives. The agent's goals and/or assumptions that form the basis for a procedure that is currently executed may be affected by a changed environment and a different set of actions may have to be performed.
- Pro–activeness: ability to exhibit goal-directed behavior by taking the initiative, responding to changes in their environment in order to satisfy their design objectives.
- Sociability: capability of interacting with other agents (software and humans) through negotiation and/or cooperation to satisfy their design objectives.

Software agents have really evolved in distributed computation paradigm. Mobile agent is a thread of control that can trigger the transfer of arbitrary code to a remote computer. Mobile agents' paradigm has several advantages: Conserving bandwidth and reducing latencies. Also, complex, efficient and robust behaviors can be realized with surprisingly little code. Mobile agents can be used to support weak clients, allow robust remote interaction, and provide scalability [2].

6. ADVANTAGES OF MAS

The advantages offered by Multi Agent System (MAS) can provide support to address a number of general data mining issues, such as:

a. The size of the data sets to be mined: Ultimately data miners wish to mine everything: text, images, video, multi-media as well as simple tabular data. DM techniques to mine tabular data sets are well established, however ever larger data sets, more complex data (images, video), and more sophisticated data formats (graphs, networks, trees, etc.) are required to be mined. The resources to process these data sets are significant; a Multi Agent Data Mining (MADM) approach may therefore provide a solution.

b. Data security and protection: The legal and commercial issues associated with the security and protection of data are becoming of increasing significance in the context of data mining. The idea of sharing data for data mining by first compiling it into a single data warehouse is often not viable, or only viable if suitable preprocessing and annoimization is first undertaken. MADM provides a mechanism to support data protection.

c. Appropriateness of DM Algorithms: An interesting observation that can be drawn from the DM research conducted to date is that for many DM tasks (for example ARM) there is little evidence of a "best" algorithm suited to all data.

Even when considering relatively straightforward tabular data, in the context of ARM, there is no single algorithm that produces the best (most representative) association rules in all cases. An agent-based process of negotiation/interaction, to agree upon the best result, seems desirable.

d. Resource intensive: Common feature of most DM tasks is that they are resource intensive and operate on large sets of data. Data sources measured in gigabytes or terabytes are quite common in DM. This has called for fast DM algorithms that can mine very large databases in a reasonable amount of time. However, despite the many algorithmic improvements proposed in many serial algorithms, the large size and dimensionality of many databases makes the DM of such databases too slow and too big to be processed using a single process. There is therefore a growing need to develop efficient parallel DM algorithms that can run on distributed systems [10].

7. CONCLUSION

Data Mining techniques are significantly used for the research work to find solutions for different types of problems. Many algorithms for DM are developed and tested with real world datasets. Most of the current generation of algorithms are computationally complex and typically require all data to be resident in main memory, which is clearly unrealistic for many genuine problems and databases. Furthermore, in certain situations, data may be inherently distributed and cannot easily be merged into a single database for a variety of reasons including security, fault tolerance, legal constraints and competitive reasons. In such cases, it may not be possible to examine all of the data at a central processing site to compute a single global result.

Many DARM algorithms are proposed in literature to solve the issues related to distributed data. Based on the various challenges and issues in DARM and Agent Mining, there is a need for the enhancement of DM and the creation of Intelligent agents. More efforts are required to develop techniques, systems, and case studies from foundational, technological, and practical perspectives.

Many of the DARM algorithms uses the datasets were generated and pre-processed in a separate off-line process. Introducing data pre-processing agents could solve the incompatible schema problem.

The existing DARM algorithms described above in the paper focuses on homogeneous horizontally partitioned data. There is a need to work on heterogeneous and dynamic data sets in a distributed environment. Along with the existing agents that are introduced in the current DARM architecture, new agents such as data preprocessing agents, fault tolerant agents and adaptive agents need to be developed that can work together effectively in a distributed environment.

REFERENCES

[1] Md. Golam Kaosar, Zhuojia Xu and Xun Yi, “Distributed Association Rule Mining with Minimum Communication Overhead,” in 8th Australasian Data Mining Conference (ausdm'09), Australia, Volume 101, 2009.

[2] Dr. Sujni. Paul, “Parallel and Distributed Data Mining,” New Fundamental Technologies in Data Mining, book edited by Kimito Funatsu, ISBN 978-953-307-547-1, January 21, 2011.

[3] Mafruz Zaman Ashrafi, David Taniar and Kate Smith, “ODAM: An Optimized Distributed Association Rule Mining,” IEEE DISTRIBUTED SYSTEMS ONLINE 1541-4922, Volume 5, No. 3, March 2004.

[4] W. Ailing, “An Improved Distributed Mining Algorithm of Association Rules,” Journal of Convergence Information Technology, Volume 6, Number 4 , April 2011.

[5] Dr(Mrs.) Sujni Paul, "An Optimized Distributed Association Rule Mining Algorithm in Parallel and Distributed Data Mining with XML Data for Improved Response time," IJCSI, volume 2, April 2010.

[6] Albashiri, Dr. Kamal Ali, “Data Partitioning and Association Rule Mining Using a Multi-Agent System,” International Journal of Engineering Science and Innovative Technology (IJESIT), Volume 2, Issue no. 5, September 2013.

[7] Shenzhi Li, Tianhao Wu and William M. Pottenger, "Distributed Higher Order Association Rule Mining Using Information Extracted from Textual Data", ACM SIKGDD Newsletter-Natural Language Processing and Text Mining, Volume 7, Issue 1, June 2005

[8] Hongjun Lu, Ling Feng and Jiawei Han, “Beyond Intra-Transaction Association Analysis:Mining Multi Dimensional Inter-Transaction Association Rules” ACM Transactions on Information Systems, Volume 18, Issue 4, October 2000.

[9] Longbing Cao, Vladimir Gorodetsky, Pericles A. Mitkas, "Agent Mining: The Synergy of Agents and Data Mining," IEEE Intelligent Systems, Volume 24, Issue 3, May 2009

[10] Albashiri, Kamal Ali, “Agent Based Data

Distribution for Parallel Association Rule Mining," INTERNATIONAL JOURNAL OF COMPUTERS, Volume 8, 2014.

AUTHOR PROFILES

Ms. Vinaya Sawant received the Masters degree in computer science from NMIMS University, in 2010. She is a research student of NMIMS University. Currently, she is an Assistant Professor at D. J. Sanghvi College of Engineering, Mumbai University.

Dr. Ketan Shah received the Ph. D degree in Information Technology from NMIMS University. Currently, he is an Associate Professor at MPSTME, Mumbai, NMIMS University.