Mobile Agent Based High Performance Distributed Data Mining Algorithm in Real Time Applications

S.KAVITHA[1]
Research Scholar, Computer Science, Research and Development Centre,
Bharathiar University, Coimbatore, Tamil Nadu, India.

and

Dr. K.V. ARUL ANANDAM[2]
Assistant Professor, Department of MCA, Govt. Thirumagal Mills College,
Vellore, Tamil Nadu, India.

ABSTRACT The massive amount of data spread in the fields of the industrial, scientific and commercial applications. Analyze large data sets maintained over geographically distributed sites by using the computational power of distributed and parallel systems is essential. A distributed computation paradigm (message passing, remote procedure calls, mobile agents) and the used integration techniques (Knowledge probing) in order to aggregate and integrate the results of the various distributed data miners. The pruning technique is an efficient method which can obtain the approximate maturity but obviously reduces the size of result of data mining process. Mobile agents act autonomously, does not waste the bandwidth because the agent migrates to the server and in addition with the parallel processes. Mostly, research has focused in the improvement of the efficiency of discovering association rules but still a gap in the practical application rules. This paper shows that DDMA-3 algorithm increases efficiency of the data extraction from distributed databases than other algorithms and shown with the experimental results. The ultimate goal of this algorithm is to extract huge amounts of data and to solve the problems associated with the model in the practical applications.

Keywords: Mobile Agent, Load Balancing, Distributed Association Algorithm, Parallel Frequent Item-sets mining,

1. Introduction

Data mining is a tool that supports research and allows new assertions to be made by disclosing previously undisclosed details in large amounts of data. Data base mining algorithm is developing fast and efficient algorithms that can deal with large volume of data because most mining algorithms perform computation over the entire database and mostly the databases are very large.Association rule mining has been a topic of active research in the field of data mining. It has two steps to complete the task which are discovering frequent sets of items in a database and subsequently extracting rules from these frequent sets of items.

S.Kavitha is currently pursuing Ph.D in Computer Science in Bharathiar Univeristy and also working as Assistant Professor, Department of Computer Applications, SRM University, Chennai, Tamil Nadu,India, E-mail: kavitha.s@ktr.srmuniv.ac.in

Dr.K.V.ArulAnandam is an Assistant Professor in computer Science in Govt. Thirumagal Mills College , India. E-mail: sathisivamkva@gmail.com.

Mobile agents are software robots (mobile code) that can move autonomously from node to node within a network, executing its operations interacting with the host environment at each node that it visits. The process migration, remote evaluation and mobile objects are the concepts of mobile agents with the improvement over Remote Procedure Call (PRC) for distributed programming. It is sent to an information source together with a query. It can actively interact with the information source that is to refine the query. It shows a reduced communication load as the interaction occurs at the same place. Mobile agent used to conserving bandwidth reducing latencies, allow robust remote interaction and provide scalability. Mobile agent paradigm is considered to be better than client server paradigm for large-scale distributed heterogeneous network environment.

The main challenges of the parallel and distributed association include work-load balancing, synchronization, communication minimization, finding good data layout, data decomposition and disk I/O minimization which is especially important for distributed association rule mining. The mobile agent technology with parallel and distributed association rules are used to meet the above challenges and also shown the improvements with the experimental results.This module contains the following steps i). Find all frequent item-sets for a predetermined supports. ii). Generate the association rules from the frequent item-sets from all sites. iii). Calculate Elapsed time with different minimum support with different algorithms. iv) Calculate Elapsed time with different database size with different algorithms.

This paper is organized as follows. Related work of the parallel and distributed algorithms surveyed and presented in section 2. Section 3 provides the research methodology about the Improved Distributed Data Mining Algorithm using mobile agent. Section 4 describes the parallel frequent item sets mining. Section 5 presents mobile agent based distributed data mining. Section 6 discussed the notations which is used for measuring the performance of the DDM-Algorithms. Section 7 reports the experimental result.Finally, summarized work of the paper is in the section 8.

2. Related Work

In the distributed environment, so many algorithms have been proposed to mine all association rules in distributed databases. DARM is the first algorithm proposed in the share nothing parallel systems and datasets are horizontally partitioned among different sites using the Apriori algorithm in parallel version. Fast Distributed Algorithm (FDM) has been proposed by Cheung et al. to mine rules from distributed data sets partitioned among different sites[3]. It prunes all infrequent local candidate sets with the help of the local support counts in each site. Then, each site broadcasts messages of candidate sets to other sites and request the support counts. Finally, finds the globally frequent item sets. ODAM algorithm proposed with the base concepts of FDM algorithm [4]. It has the advantage of greater message optimization and performance enhancement to FDM. Distributed Decision Miner (DDM) is proposed by Assaf Schuster and his colleagues which generates the rules that have confidence above the threshold level without generating a rule's exact confidence. Distributed Sampling (D-Sampling) algorithm is proposed by Assaf et al which is an extension of the sequential sampling in distributed databases[5][6].

3. Research Methodology

Let DB be a database with D transactions. In a distributed system, there are n sites (S_1,S_2,…,S_n). The database partitioned over the n sites into DB_1, DB_2, … ,DB_n where $DB=\cup DB_i$, i=1 to n , $DB_i \Omega DB_j = \phi$ for i=j. Let the size of the partitions DB_i be D_i, for i=1, … , n. The support of an itemset S is defined as the fraction of total transactions that contain S. An itemset is called frequent if its support is above a user specified minimum support threshold t. Let X.sup and $X.sup_i$ be the support counts of an itemset X in DB and DB_i respectively. Let IDBI be the number of transactions in global database DB and IDB_iIbe the number of transactions in local database DB_i.X.sup is called the global support count and $X.sup_i$the local support count of X at site S_i. For a given minimum support threshold minsup, X is globally

frequent if X.sup ≥ minsup X D ; correspondingly , X is locally frequent at site S_i, if X.supi ≥ minsup X D_i. GFI denotes the globally frequent itemsets in DB, LFI_i denotes the locally frequent itemsets in DB_i. The goal of parallel frequent itemsets mining algorithm is to find the globally frequent itemsets GFI.

Definition 1. X is globally large, if X.sup ≥ Minsup * IDBI.

***Definition 2*.** X is locally large in site S_i, if X.supi ≥Minsup * IDB_iI.

***Lemma 1*.** If an itemsetX is globally large, there exists a site S_i (1≤i≤n), such that *X* and all its subsets are locally frequent at site S_i.

Proof: If *X* is not locally large at any site, then $X.sup_i$< s x D for all i=1,…,n. Therefore, $X.sup_i$< s x D and X cannot be globally large. By contradiction, *X* must be locally large at some site S_i and hence *X* is locally large at S_i. Consequently, all the subsets of *X* must also be locally large at S_i.

***Lemma 2*.** If the set of the globally large itemsets is GFI, the set of the local large itemsets at site S_i is LFI_i, then GFI $\subseteq \cup LFI_{i,\ i=1\ to\ n}$.

Proof: If the sets are globally large itemsets, it should be locally large itemsets.

***Lemma 3*.** For an itemset X, if the set of sites at which *X* is large noted as *X* in frequent sites, the set of sites at which *X* is not large as *X* in infrequent sites.

Proof: It is clear that *X* in frequent sites $\cup X$ in infrequent sites = $\{S_1,S_2,\ldots,S_n\}$ and *X* in frequent sites ΩX in infrequent sites = ϕ.

***Lemma 4*.** If *X*.Minsup ≥ sup xIDBI, then *X* is globally large itemset.

Proof: If *X* is globally large itemset, it should satisfy the condition that the minimum support of *X* should be greater or equal to the multiplying of the support count with the number of transactions in the given database.

***Lemma 5*.** If *X*.maxsup< sup xIDBI, then *X* cannot be globally large itemset.

Proof: If *X* cannot be a globally large itemset, it should satisfy the condition that the minimum support of *X* should be less than the multiplying of the support count with the number of transactions in the given database.

4. Parallel Frequent Item set Mining

To improve frequent pattern mining performance, many researchers distribute the mining computation over more than one processor. The transactional database (D) divides equally among the available processors (P). Each processor gets N/P transactions ($D_{N/P}$) where N and P are the total number of transactions in the database and the number of processors available respectively. Each processor counts the frequency of each item x using its local data partition all worker processors sends the local count to master processor. The master processor collects all such items and combines them to generate a global count. The advantage of the computation power of parallel machines provides, reduces significantly, and requires scanning the dataset and generated Frequent Patterns [2]. The following steps shows the Parallel Execution[2]

1) Server starts processing by splitting the data.
2) Server sends the data and control to each node in distributed environment.

3) Each node generates patterns i.e., frequent item-sets as per their sets and support given.
4) Each node also generates association rules from the frequent item-sets as per the confidence given.
5) The nodes return the control to the server.

5. Mobile Agent Based Distributed Data Mining Algorithms

The mobile agent technology is an efficient tool for searching and retrieving information and also achieve the efforts for reducing the connection time, the communication time and the informational retrieval time [1]. In the network perspective, the mobile agent technology can reduce the network traffic, overcome network latencies and the enhance robustness and fault-tolerant capabilities of distributed applications. Because of these advantages, the mobile agent technology is applied in the distributed data mining and also parallel concepts used for speed up the process of mining[9].The following is the basics of the DDM algorithm of using mobile agent

- To find the local knowledge from the distributed sites.
- To integrate the local knowledge in oder to find global knowledge.
- To check the quality in the global knowledge.

The aim of the algorithm is to reduce the scan of the database and decrease the number of candidate of global frequent itemsets in the distributed database using mobile agents. So for, this algorithm uses the less communication overhead and improves the efficiency of mining global frequent itemsets. It is proved from the experimental results.

5.1 Notations

The following notations are used in the DDM-algoritms.

- DB - Database.
- D - Number of Transaction.
- n - Number of sites (S1, S2, … Sn).
- DBi - Distributed Data sets at Si ,DB =U DBi, i= 1 to n.
- X.Sup - Support count of a X at DB –Global.
- X.Supi - Support count of a X at DBi –Local.
- Minsup-Minimum support threshold.
- GFI - Global Frequent Item Set.
- CGFI - Candidate Global Frequent Item Set.
- X - Global Frequent Item iff, X.Sup>= minsup * D.
- X - Local Frequent Item iff, X.supi>= minsup * Di.
- LFI I - Local Frequent Item set at site-i.
- PGFI - Possible Global Frequent Item Sets- These are item sets at sites-i, which are not part of LFI i, but by adding these count at central place converts CGFI to GFI [7].

5.2 DDM Algorithm (DDMAlgorithm-1)

The goal of this algorithm is to minimize scans of the database and data is easy to use and update. Moreover it makes each processor to process independently and decrease the number of candidate global frequent item sets according to the relation between local frequent item sets and global frequent sets. It has two steps as follows [8]

1. Mining Local Frequent Item Sets (LFI) at each site in parallel and send them to central site to calculate Global Frequent Item Stets.
2. Central site calculates the Candidate Global Frequent Item Sets – CGFI and send them to all sites. Each local site computers counts the CGFI and sends them back to central site to complete the process of finding GFI.

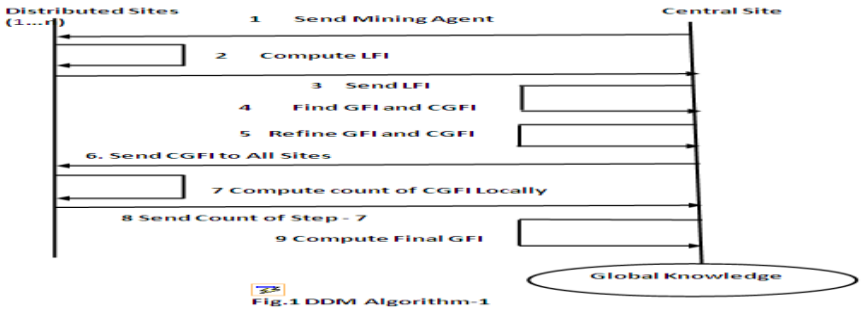

Fig.1 DDM Algorithm-1

Input : Distributed-Data-Set DB_ii=1 to n, minsup, Distributed sites' address.

Output : Global Frequent Item set – GFI

1. Send Mining Agent (MA) to all sites with support value.
2. Each Cooperative MA_i, Computes LFI $_i$ in parallel.
3. Send LFI_i(i=1 to n) to central site.
4. Compute GFI & CGFI: GFI=∩ LFI $_I$, CGFI= Ù LFI $_i$ - ∩ LFI $_I$, $_{i=1 \text{ to } n}$.
5. Add an item set to a GFI, if its count in frequent sites is greater or equal to minimum support value.

```
For all X ε CGFI do
{
        If Σ i= 1 to n X.Sup i >= minsupx D Then
        {
                GFI = GFI Ù {X}; CGFI = CGFI - {X};
        }
 }
```

6. Send CGFI to each site $S_{i\ ,i=1 \text{ to } n}$.
7. Compute counts of any item set X.(X ε CGFI) in infrequent sites.

```
For i = 1 to n do
{
        Search X at site Si;
        Get X.Supi and send to central site.
}
```

8. Send count of each CGFI to central site.
9. At Central site: Compute Global Counts of each item set in CGFI

For all X ε CGFI do

{

If *X*.Sup = Σ X.Sup $_{I, I= 1 \text{ to } n}$>= minsupxD then

{

GFI = GFI Ù {X};

}

}

5.3 DDM Algorithm (DDMAlgorithm-2)

The basic objective is to reduce the time required to compute GFI. This algorithm performs two tasks parallel.
1. Local sites send LFIs to central site and also to all their neighbors.
2. Calculation of GFI/CGFI at central site and counts of CGFI at local sites is done as overlapped operation. That is, local sites need not wait for central site to send CGFI. Thus total time taken is reduced drastically.

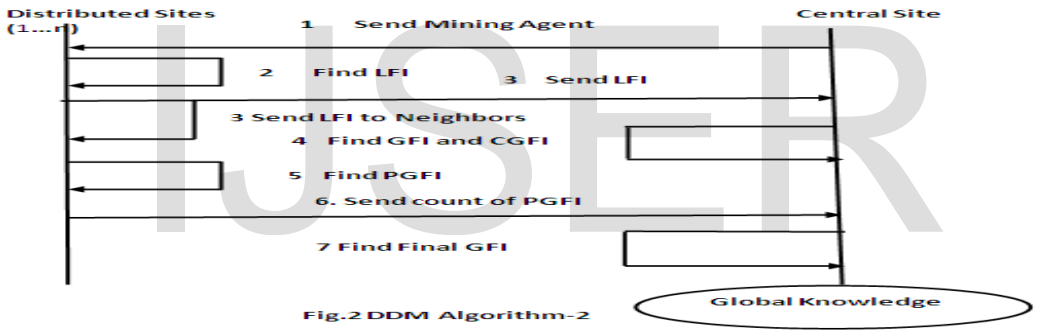

Fig.2 DDM Algorithm-2

Input: Distributed-Data-Set $DB_{i, i=1 \text{ to } n}$, Minsup.

Output: Global Frequent Item set – GFI.

1. Send Mining Agent (MA) to all sites.
   For I= 1 to n do
   {
   MA.send (Location = I, S=Support, Addresses of all distributed sites);
   }
2. Each Cooperative MA_i Computes LFI_{I} in parallel.
3. Send LFI to central site and also to all its neighbors.
4. Compute GFI & CGFI at central site.
   GFI=∩ $LFI_{I, i=1 \text{ to } n}$; CGFI= Ù LFI_{i} - ∩ $LFI_{I, I = 1 \text{ to } n}$.
5. Calculate PGFI and their count at each site.
   PGFI j=all sites= All Item Sets at site-j ∩ $LFI_{i, i=1 \text{ to } n, i<>j}$
6. Send count of $PGFI_{i, i=1 \text{ to } n}$ to central site from each infrequent site.
   Note: Step-4 and Step-5, 6 are performed in parallel.
7. Calculate GFI at central site using PGFI count.
   for all X ε CGFI do
   {

If X.Sup = Σ $X.Sup_{i,i= 1 to n}$>= Minsupx D then
{
GFI = GFI Ù {X};
}
}

5.4 Improved DDM Algorithm (DDMAlgorithm-3)

The new algorithm performs the following tasks parallely.

Mining Local Frequent Item sets (LFI) and Possible Global Frequent Item Sets with count which is not part of LFI at

1. each site in parallel and send them to central site for calculation Global Frequent Item sets.
2. Calculation of Gobal Frequent Item Sets (GFI)/Candidate Global Frequent Item Sets (CGFI) and also find final Gobal Frequent Item Sets at cental site in parallel. Central site need not wait for PGFI count for GFI calculation. So, Total time taken is reduced [7].

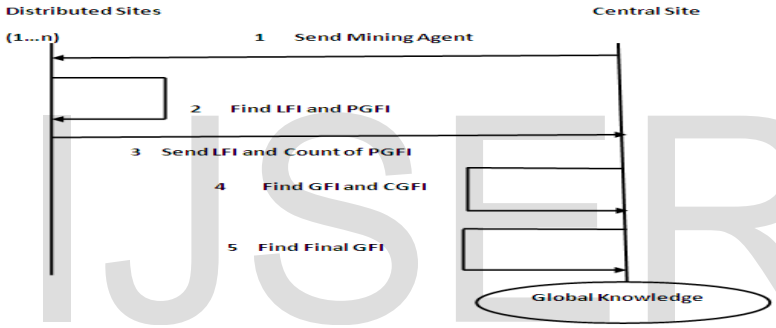

Fig.3 DDM Algorithm-3

Input : Distributed-Data-Set DBi, i=1 to n, Minsup, Distributed sites' address.

Output : Global Frequent Item set – GFI.

1. Send Mining Agent (MA) to all sites.
   for I=1 to n do
   { MA.send(Location=I,S=Support,Address of all distributed sites);
   }
2. Each Cooperative MA_i Computes LFI_I in parallel, PGFI and their count at each site .
   for I=1 to n do
   {If frequent present then
   Compute LFI_I
   else
   PGFI $_{j=all\ sites}$ = All Item Sets at site-j ∩$LFI_{i,\ i=1\ to\ n,\ i<>j}$

   }
3. Send LFI to central site and also send count of $PGFI_{i,\ i=1\ to\ n,}$ to central site from each infrequent site.
4. Compute GFI and CGFI at central site.GFI=∩ $LFI_{i,i=1\ to\ n}$; CGFI= ∪ LFI_i - ∩ $LFI_{i\ ,i=1\ to\ n.}$
5. Calculate GFI at central site using PGFI count.

for all X ε CGFI do

{

If X.Sup= Σ X.Sup $_{i,i=1 \text{ to } n}$>=MinSup*D then

{

GFI=GFI ∪ {X};

}

}

Note : Step 4 and step 5 are performed parallel.

The DDM Algorithm-3 has improved because of the implementation of steps reduction methods and parallel programming when compared to the existing algorithms. This will lead to the reduction of message passing among the nodes. It will increase the execution time and decrease the communication overhead.

6. Performance Measurement

The followings are the notations used for measuring the performance of the above DDM Algorithms and also shows the differentiation among the DDM-Algorithms[10].

- Ts – Time required sending 'minsup' from main site to all distributed sites.
- Tc-LFI – Time required calculating LFI at all distribute sites.
- Ts-LFI-m – Time required sending LFI from all distributed sites to central site.
- Ts-LFI-n – Time required sending LFI from all distributed sites to their neighbors.
- Tc-C/GFI-m –Time required to find GFI and CGFI at central site.
- Ts-CGFI-ds – Time required sending CGFI from central site to all distributed sites.
- Ts-c-CGFI-m – Time required finding count of CGFI at each distributed sites and sending to central site.
- Ts-PGFI-m – Time required finding and sending PGFI and its count to central site.
- Tco-P/CGFI – Time required converting CGFI to GFI using count of PGFI received from all distributed sites.
- T1 ,T2& T3 – Time required to find GFI at central site using existing DDM Algorithm-1 , DDM Algorithm-2 and proposed methods respectively.

Total time required for calculating GFI using DDM algorithm-1, DDM algorithm-2 and DDM-Algorithm-3 are T1, T2 and T3 respectively.

$$T1 = Ts + Tc\text{-}LFI + Ts\text{-}LFI\text{-}m + Tc\text{-}C/GFI\text{-}m + Ts\text{-}CGFI\text{-}ds + Ts\text{-}c\text{-}CGFI\text{-}m + Tco\text{-}P/CGFI\text{-} \quad (1)$$

$$T2 = Ts + Tc\text{-}LFI + Ts\text{-}LFI\text{-}m + Tc\text{-}C/GFI\text{-}m + Tco\text{-}P/CGFI \quad - \quad (2)$$

$$T3 = Ts + Tc\text{-}LFI + Ts\text{-}LFI\text{-}m + Tco\text{-}P/CGFI \quad - \quad (3)$$

From the above formula, it gives clear solution that T3<T2<T1 according to less synchronization steps and parallelism. It is evident that DDM-Algorithm-3 has less time requirement for obtaining the global knowledge.

7. Experimental Results

These experimental results have been taken from the real time data sets of super market using MATLAB programming. The following figures (1 and 2) shows that the comparison of the different algorithms in terms of performance of speed and scale-up.If the percentage of the support threshold is

increased, the elapsed time in secondsof the algorithms decreased. The DDMA-3 has very less elapsed time in secondsthat shown in the fig.1 and the performance scalability is more when compared to the existing algorithmsthat shown in the fig.2. From that the DDMA-3 is improved algorithm than the DDMA-1 and DDMA-2.

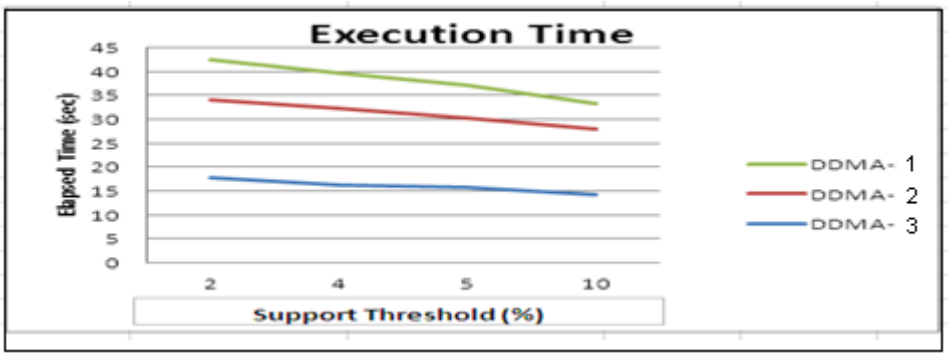

Fig. 1 Comparison of different algorithms DDMA-1, DDMA-2 and DDMA-3

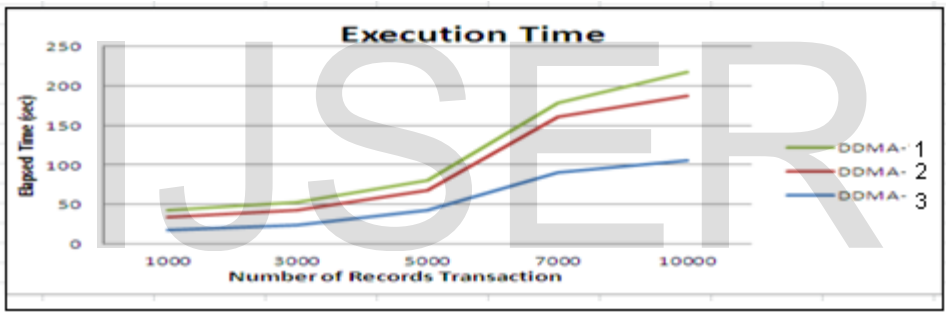

Fig.2 Performance of Scale-Up

8. Conclusion

Association rules mining in distributed database using mobile agent technology is an important aspect in the field of data mining. Our algorithm showed that it can reduce the computation overhead of discovering frequent item sets in a distributed environment and also scalable in terms of the required communication and computation costs. This algorithm can provide valuable information for decision-making and prove with experimental result.

Reference

[1]. Manvi.S.S, Venkataram.P, “Applications of agent technology in communication: a review”, Computer Communication”, (2004), 1493-1508, Elsevier, Science Direct.
[2]. Agrawal, R. and Shafer,J. (1996). “Parallel mining of association rules”. IEEE Transaction on Knowledge and Data Engineering, Vol.*, No.6, pp.962-969.
[3]. Cheung, D.W. et al. (1996). “A fast distributed algorithm for mining association rules”, Proc. Parallel and Distributed Information Systems, IEEE CS Press, pp. 31-42.
[4]. Ashrafi, M. Z., Taniar, D. and Smith, K. (2004). ODAM: An Optimized distributed association rule mining algorithm. IEEE distributed systems online, Vol. 05, No.3.

[5]. Schuster, A. and Wolf, R. (2001). “Communication-efficient distributed mining of association rules”. Proc. ACM SIGMOD International Conference on Management of Data, ACM Press, pp. 473-484.
[6]. Schuster, A., Wolf, R. and Trock, D. (2005). “A high-performance distributed algorithm for mining association rules”. Knowledge and Information Systems (KAIS) Journal, Vol.7, No.4.
[7]. Kavitha.S, and Dr.KV.ArulAnandam, “Research on improved distributed data mining algorithm using mobile agent framework”, Intl. j. of scientific and Engineering Research (IJSER), vol.4, issue 6,(2013),pp.740-744, ISSN 2229-5518.
[8]. Yun-Lan Wang, Zeng-Zhi Li, Hai-Ping Zhu, “Mobile-Agent-Based Distributed and Incremental Techniques for Association Rules”.Procceedings of Second Int. Conf. on Machine Learning and Cybernetics, Xi’an, (2003), IEEE.
[9]. J.Han, J.Pei, Y.Yin, R.Mao, “Mining frequent patterns without candidate generation: a frequent-pattern tree approach”, Journal of Data Mining and Knowledge and Grid, (2009), pp.128-135.
[10]. U.P.Kulkarani, P.D.Desai, Tanveer Ahmed, J.V.Vadavi and A.R.Yardi, “Mobile Agent Based Dustributed Mining”, International Conference on Computational Intelligence and Multimedia Applications (2007).