Mobile Agent Based Distributed Data Mining

U.P.Kulkarni {1}, P.D.Desai {2}, Tanveer Ahmed {3}, J.V.Vadavi {4}, A.R.Yardi {5}

{1, 2, 3, 4} Faculty, Department of Computer Science and Engineering,
SDM College of Engineering and Technology, Dharwad, India.
{5} Dy.Director, Walchand College of Engineering and Technology, Sangli, India.

prof_ard@yahoo.com

Abstract

The analysis of large datasets has become an important tool in understanding complex systems in areas such as economics, business, science and engineering. Such datasets are often collected geographically distributed way and cannot in practice be gathered in to a single repository. Applications that work with such datasets cannot control most aspects of the data’s partitioning and arrangements. So far, attention in data mining process has always focused on extracting information from data physically located at one central site and they often do not consider the resource constraints of distributed and mobile environments. Few attempts were also made in parallel data mining. However most real life applications rely on data distributed in several locations. As a consequence both new architectures and new algorithms are needed. In this paper author proposes a method that explores the capabilities of mobile agents to build an appropriate framework and an algorithm that better suits the Distributed Data Mining applications. It also makes the performance analysis and comparison with the existing such method [4].

1. Introduction

Data mining is the new technology of discovering the meaningful information from the data repository, which is widely used in almost all domains, which includes: finance, insurance, process control, quality supervising, engineering and scientific data analysis etc. Recently, mining of databases has attracted a growing amount of attention in database communities due to its wide applicability in retail industries in improving marketing strategies. Analysis of past transaction data can provide very valuable information on customer behavior and business decisions. The amount of data stored grows twice as fast as the speed of the fastest processor available to analyze it. The problem of extracting the knowledge is harder for large datasets due to the following properties:

- Datasets are distributed geographically so pieces of the same logical datasets are physically located far away from each other.
- Datasets are immovable in practice.

Because of these properties, algorithms to compute with large datasets cannot assume or control the partitioned structure, the sizes, and the locations of the pieces of the datasets and must take account of the latencies and bandwidth required to move data among the pieces.

2. Related Work

Several Distributed Data Mining (DDM) systems using client server model, agent-based model and grid-based architectures including the parallel mining on clusters were proposed. DDM system based on client server model includes: Decision Center [1], IntelliMiner [2]. All these are based on the technologies like: CORBA, DCOM, EJB, RMI, JDBC etc. Agent-based DDM [3] includes JAM, BODHI, and DAME. The paper [5] highlights the capabilities of Mobile Agents in building the various distributed applications, particularly focusing on the
Distributed Data Mining. But the issue of consolidating the knowledge collected from various distributed sites is not being addressed.

These existing approaches for DDM suffer from one or more of the following limitations [3].
- Lack of coordination among the distributed sites while generating the local knowledge, affects the quality of Global knowledge.
- Non-flexibility in addition of new algorithm to its knowledge base.
- Capability to dynamically discover data sites based on user requirements.

The paper [4] proposes the parallel algorithm for mining frequent item sets, which uses two steps with some scope for further improvements. This paper suggests the improvements over the method proposed by You-Lin Ruan et al [4] and uses the advantages of mobile agents [5] over client server based approaches in terms of better bandwidth usage and network latency.

3. Proposed Concept

To overcome all these, a mobile agent based DDM has been proposed. Here knowledge from distributed sites are extracted in the from of association rules. Based on the types of values, the association rules can be classified into two categories:
- Boolean Association Rules: Keyboard \( \Rightarrow \) Mouse [Support = 6%, confidence = 70%]
- Quantitative Association Rules: (Age = 26 \ldots 30) \( \Rightarrow \) (Cars = 1, 2) [Support = 3%, confidence = 36%]

3.1. Basics

Generally the DDM process consists of following steps:
1. Generate the knowledge locally at each distributed sites.
2. Integrate the local distributed knowledge to get global knowledge.
3. Quality check on the global model.

The notations used in the methodology proposed are as below:
- DB \( \Rightarrow \) Database.
- D \( \Rightarrow \) Number of Transaction.
- n \( \Rightarrow \) Number of sites ( S1, S2, \ldots Sn ).
- DBi \( \Rightarrow \) Distributed Data sets at Si ,
  \( DB = \bigcup DBi, \quad i = 1 \ldots n \).
- X.Sup \( \Rightarrow \) Support count of a X at DB –Global.
- X.Supi \( \Rightarrow \) Support count of a X at DBi –Local.
- Minsup \( \Rightarrow \) Minimum support threshold.
- GFI \( \Rightarrow \) Global Frequent Item Set.
- CGFI \( \Rightarrow \) Candidate Global Frequent Item Set.
- X \( \Rightarrow \) Global Frequent Item iff, X.Sup >= minsup * D.
- X \( \Rightarrow \) Local Frequent Item iff, X.supi >= minsup * Di, \( i \).
- LFIi \( \Rightarrow \) Local Frequent Item set at site-i.
- PGFI \( \Rightarrow \) Possible Global Frequent Item Sets- These are item sets at sites-i, which are not part of LFIi, but by adding these count at central place converts CGFI to GFI.

3.2. Methodology

The method proposed by You-lin Ruan et al[4] which 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 item sets. It basically uses 2 steps.

1. Mining Local Frequent Item sets (LFI) at each site in parallel and send them to central site to calculate Global Frequent Item sets.
2. Central site calculates the Candidate Global Frequent Item sets-CGFI and send them to all sites. Each local site computes the count of such CGFI and sends them back to central site to complete the process of finding GFI.

**DDM Algorithm-1:**

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 with support value.
2. Each Cooperative MAi, Computes LFIi in parallel.
3. Send LFIi (i=1 to n) to central site.
4. Compute GFI & CGFI: GFI= ∩ LFIi, CGFI= û LFIi - ∩ LFIi, i=1 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 ∈ CGFI do 
   
   \[ \text{If } \sum_{i=1}^{n} X.\text{Sup}_i \geq \text{Minsup} \times D \text{ Then} \]
   
   \[ \text{GFI = GFI } \cup \{X\}; \quad \text{CGFI = CGFI - } \{X\}; \]

6. Send CGFI to each site Si, i=1 to n.
7. Compute counts of any item set X
   
   (X ∈ 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 ∈ CGFI do 
   
   \[ \text{If } X.\text{Sup} = \sum_{i=1}^{n} X.\text{Sup}_i, i=1 \rightarrow n \geq \text{Minsup} \times D \text{ then} \]
   
   \[ \text{GFI = GFI } \cup \{X\} \]

**Proposed Algorithm**

The proposed algorithm is an improvement over above DDM approach. The basic objective is to reduce the time required to compute GFI. The proposed 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
a overlapped operation. That is, local sites need not wait for central site to send CGFI. Thus total time taken is reduced drastically.

**DDM Algorithm-1:**

- **Input:** Distributed-Data-Set $DB_i$, $i=1$ 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.
   
   $GFI = \bigcap_{i=1}^{n} LFI_i$ ; $CGFI = \bigcup_{i=1}^{n} LFI_i - \bigcap_{i=1}^{n} LFI_i$.

5. Calculate PGFI and their count at each site.
   
   $PGFI_{j=\text{all sites}} = \text{All Item Sets at site-j} \cap LFI_{i}, i=1 \text{ to } n, i<>j$

**Fig-1: DDM Algorithm-1.**

**DDM Algorithm-2:**

- **Input:** Distributed-Data-Set $DB_i$, $i=1$ 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.
   
   $GFI = \bigcap_{i=1}^{n} LFI_i$ ; $CGFI = \bigcup_{i=1}^{n} LFI_i - \bigcap_{i=1}^{n} LFI_i$.

5. Calculate PGFI and their count at each site.
   
   $PGFI_{j=\text{all sites}} = \text{All Item Sets at site-j} \cap LFI_{i}, i=1 \text{ to } n, i<>j$
6. Send count of PGFI\textsubscript{i, i=1 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 \in CGFI do
   \{
   \text{If X.Sup = } \sum X.Sup \mid i=1 \text{ to } n \geq \text{MinSup} * D \text{ then}
   \{
   \text{GFI = GFI } \cup \{X\}
   \}
   \}

Note: step-6 of DDM algorithm-1 is totally eliminated in DDM algorithm-2.

3.3 Performance Measurement

The proposed algorithm uses the basic approach of finding GFI as proposed by You-Lin Ruan et al, including the terminologies used and adds overlapped activity, thereby reducing the time taken to mine geographically distributed data.

Notations used for timing analysis are:

- **Ts** – Time required to send ‘minsup’ from main site to all distributed sites.
- **Tc-LFI** – Time required to calculate LFI at all distribute sites.
- **Ts-LFI-m** – Time required to send LFI from all distributed sites to central site.
- **Ts-LFI-n** – Time required to send 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 to send CGFI from central site to all distributed sites.
• Ts-c-CGFI-m – Time required to find count of CGFI at each distributed sites and send to central site.
• Ts-PGFI-m – Time required to find and send PGFI and its count to central site.
• Tco-P/CGFI – Time required to convert CGFI to GFI using count of PGFI received from all distributed sites.
• T1 & T2 – Time required to find GFI at central site using existing and proposed methods respectively.

Total time required for calculating GFI using DDM algorithm-1 is:
T1 = Ts + Tc-LFI + Ts-LFI-m + Tc-C/GFI-m + Ts-CGFI-ds + Ts-c-CGFI-m + Tco-P/CGFI

The proposed DDM algorithm-2 does not use Ts-CGFI-ds and Ts-c-CGFI-m as central site is not sending CGFI. Local sites get this information from its neighbors. Thereby communication time is reduced, which results in reduction of total time to find GFI. Thus total time taken by this DDM algorithm-2 is,

T2 = Ts + Tc-LFI + Ts-LFI-m + Tc-C/GFI-m + Tco-P/CGFI

It is clear that T2<T1 due to less synchronization steps and parallelism, which is also evident from the experimental results conducted in Local Area Network, as shown in figure-3. The Implementation of the proposed system over internet is in progress. Figure-4 shows the GUI of the system implemented.

4. Conclusion

This work aims at providing solutions to the issue of knowledge consolidation with less communication overhead due to minimum information exchange by overlapped operations, thereby improving the efficiency of DDM. Percentage of saving in processing time increases with the distance between the central site and distributed business sites.

![Performance Comparison](image)
5. Acknowledgement

Authors sincerely acknowledge Franz Coenen of Liver Pool University, for using his Apriori implementation logic, dataset generator and SDMCET-CSE/ISE-2007 group consisting of: Sudheer Havanur, Sandeep Wajpey, Praveen Lokur, Rayan Rodrigues, Raksha and Preeti for their programming support.

6. References


