Department of Information Technology

technical report NUIG-IT-170503

A Multi-Agent System for Context-based Distributed Data Mining

Yan Xing (NUI, Galway)
Michael G. Madden (NUI, Galway)
Jim Duggan (NUI, Galway)
Gerard Lyons (NUI, Galway)

A Multi-Agent System for Context-based Distributed Data Mining

Yan Xing, Michael G. Madden, Jim Duggan and Gerard J. Lyons.

Abstract—The structure of virtual organizations presents a significant challenge to the traditional methods of distributed data mining that are based on ensemble learning. The heterogeneity that arises from different contexts mitigates against the chance that preconditions for algorithms' success are satisfied. This paper describes an approach that aims to resolve this issue. Focusing on a key business problem – the prediction of customer behaviour – it presents a distributed multi-agent framework that deals with context heterogeneity via hierarchical modeling. The main elements of this work are to (1) provide a solution to the contextual heterogeneity problem in distributed data mining and (2) design and implement a hybrid distributed system for the proposed distributed data mining approach.

Index Terms—Multi-Agent Systems, Distributed Data Mining, Hierarchical Modelling, Virtual Organizations

I. Introduction

The aim of this research is to extend distributed data mining techniques so that they can be successfully applied to common business problems that arise in a virtual organization, which has been described as "a temporary network of companies that come together quickly to exploit fast changing opportunities… with each partner contributing what it's best at" [1]. A feature of such "temporary networks of companies" is that they are likely to have significant contextual and implementation differences, and that their desire to cooperate with each other will be contingent on not having to expose their detailed business data. The most recent literature in distributed data mining highlights that many of the key algorithms are designed based on ensemble learning. As discussed below, while such approaches can yield proven results, a problem arises when the contextual heterogeneity exists across the data sources.

Paper submitted on January 1st, 2003.

The support of the Informatics Research Initiative of Enterprise Ireland is gratefully acknowledged.

Yan Xing is with the Enterprise Computing Research Group, National University of Ireland, Galway. (phone: 353-91-524411; fax: 353-91-750501; e-mail: yan.xing@nuigalway.ie).

Michael G. Madden is with the Enterprise Computing Research Group, National University of Ireland, Galway. (e-mail: michael.madden@nuigalway.ie).

Jim Duggan is with the Enterprise Computing Research Group, National University of Ireland, Galway. (e-mail: jim.duggan@nuigalway.ie).

Gerard J. Lyons is with the Enterprise Computing Research Group, National University of Ireland, Galway. (e-mail: gerard.lyons@nuigalway.ie).

II. Review of Distributed Data Mining

Distributed data mining (DDM) accepts that data may be inherently distributed among different loosely coupled sites that are connected by a network, and that the sites may have heterogeneous data. It offers techniques to discover new knowledge through distributed data analysis and modeling using minimal communication of data [2].

A. Related Work

Distributed data mining research classifies existing DDM approaches according to how the data are distributed [3, 4]. Within the context of relational database schemata, data distribution is classified into two main kinds: homogeneous schemata and heterogeneous schemata. Homogeneous schemata means that the same set of attributes can be obtained across distributed data sites, while heterogeneous schemata refers to different sets of attributes across distributed databases, usually being restricted to a simple scenario where every participating table shares a common key column that links corresponding rows across the tables. Most existing DDM work considers homogeneous schemata and some deals with heterogeneous schemata.

In the scenario being considered in this paper, the same set of attributes can be obtained from distributed databases, so here we review only the approaches for distributed data with homogeneous schemata.

Most DDM algorithms for regression or classification have their foundations in ensemble learning [4]. One notable successful adoption of ensemble learning in a distributed scenario is the meta-learning framework. It offers a way to mine classifiers from homogeneously distributed data. In this approach, supervised learning techniques are first used to build classifiers at local data sites; then meta-level classifiers are learned from a data set generated using the locally learned concepts. Meta-learning follows three main steps. The first is to generate base classifiers at each site using a classifier learning algorithms. The second step is to collect the base

classifiers at a central site, and produce meta-level data from a separate validation set and predictions generated by the base classifier on it. The third step is to generate the final classifier (meta-classifier) from meta-level data via a combiner or an arbiter.

Other two well-known techniques within the meta-learning paradigm are distributed learning with knowledge probing and distributed learning in data fusion systems [3]. Both of these approaches produce a final meta-level descriptive model as its global classifier. In contrast, the final classifier of the meta-learning approach includes both a meta-classifier and local (base) models.

Since all the above techniques have their foundations in ensemble learning, they inherently adopt its requirements for success, which are discussed in the following sub-section.

One notable DDM technique that is outside of the meta-learning framework is Páircéir's work [5]. This approach uses statistical hierarchical modeling to discover multi-level association rules and exceptions over dispersedly hierarchical data. It is not based on ensemble learning. We mention this approach because our approach will also use statistical hierarchical modeling; however, our target is to address context heterogeneity.

B. Condition for Success

Ensemble learning is a well-established approach [6]. Its idea is to integrate multiple models for the same problem. Its main goal is to obtain a better composite global model, with more accurate and reliable estimates or decisions.

A learning set of L consists of data $\{(y_n, \mathrm{x}_n),\ n = 1,..., N\}$ where the y s are either class labels or a numerical response. A learning procedure uses this learning set to form a predictor $j(\mathrm{x}, \mathrm{L})$. If there is a sequence of learning sets $\{\mathrm{L}_k,\ k = 1,..., K\}$ each consisting of N_k independent observations from the same underling distribution P as L , the mission of ensemble learning is to use the $\{\mathrm{L}_k,\ k = 1,..., K\}$ to get a better predictor than the single learning set predictor $j(\mathrm{x}, \mathrm{L})$. When y is numerical, it is a regression problem. When y represents a class $j_c \in \{1,..., J\}$, it is a classification problem. The simplest ensemble combining strategy is the combiner, where the ensemble is determined solely from the outputs of all the individual learners via (weighted) averaging or majority vote. Arbitration is a more complex combining method [6, 7].

A necessary and sufficient condition for an ensemble of learners to be more accurate than the average performance of its individual members is that the learners are accurate and diverse [8, 9].

Meta-learning based DDM techniques directly extend ensemble approach to distributed scenarios [4]. Different models are generated at different sites and ultimately aggregated using ensemble combining strategies. Thus the requirements for success are the same: base learners generated at different sites must be accurate and diverse. In fact, the requirement of diverse base models is always satisfied in distributed scenarios. The reason is that the learning sets $\{\mathrm{L}_k,\ k = 1,..., K\}$ locating at different sites naturally exist and are different. So base learners produced at different sites are always diverse even with the same base learning algorithms. The requirement of accurate base learners means that each base learner is a representative of the unknown and real model based on the whole population. This implicitly requires that the learning sets $\{\mathrm{L}_k,\ k = 1,..., K\}$ are homogeneously drawn from the underling population and the difference between them is only random sampling error. From a statistical prospective, data are identical and independent distributed (IID) [10, 11] with mean μ and variance σ^2, i.e.

$$y_{ik} \overset{IID}{\sim} P(\mu, \sigma^2) \qquad k = 1,..., K \tag{1}$$

In order to obtain success, existing meta-learning based approaches also implicitly assume that the distributed data are IID [12]. In the context of virtual organizations, the validity of this assumption needs to be further explored.

C. Limitation of the State of the Art for Our Scenario

We apply the concept of a virtual organization to a loosely coupled business of retailers who may decide to share information for their mutual benefit, where within this loosely coupled organization, each shop stores detailed consumer and transaction data at its own site. We will perform consumer behavior analysis on the entire virtual organization by learning consumer shopping patterns based on data located at various dispersed companies. The business situation is that there is a business with multiple shops each mailing different catalogs to a unified consumer base. The shops sell sets of products with some overlap. After 36 months business, the organization would like to build consumer behavior models (for example, how much money a consumer spends on average) over data located at different shops, and then use the model to predict its consumers' future behavior.

Is it appropriate for data distributed in a virtual organization such as our scenario to be assumed IID? Or can existing meta-learning based DDM techniques be successfully used in our scenario? We discuss this question from four aspects.

Firstly, background of data generation: in original applications of ensemble learning, the size of the learning data set L is limited. In order to get more accurate results, the sequence of learning sets $\{\mathrm{L}_k,\ k = 1,..., K\}$ is generated by random sampling over L . Thus each element of $\{\mathrm{L}_k,\ k = 1,..., K\}$ can be regarded as homogeneously drawn from the underlying distribution P as L because the difference between them is only the random sampling error.

So it is reasonable to assume the data are IID. But in distributed scenarios, different sites store their private data locally, that is, if we have K sites, then the sequence of learning sets $\{L_k,\ k = 1,..., K\}$ naturally exist. The overall learning set is assumed obtained by $L = \bigcup L_k$. In this case, if and only if all the sites are completely identical, $\{L_k\}$ can be viewed homogeneously drawn from the underlying distribution P. Existing DDM approaches assume that all the sites are homogeneous and context heterogeneity is negligible. But unfortunately, it is difficult for us to assume that various companies within a virtual organization are homogeneous. In fact, heterogeneity of different sites is often the rule rather than the exception, and frequently the available predictor variables do not "explain" this heterogeneity sufficiently [12]. In our application scenario, different shops adopt different business policies, sell different products and have different price standards. All these context heterogeneities definitely influence consumers' purchase behavior. Consumers within a shop behave more similarly to each other than those belonging to different shops. So in our scenario the distribution is a part of semantics [12]. It is not appropriate for us to neglect the inherent context heterogeneity and assume that distributed data are IID.

Secondly, accuracy of base learners: since there is inherent context heterogeneity, it is also difficult for us to make sure that each base learner is a representative of the real behavior model of all consumers from various shops, even though the base learner is accurate at the site where it is created. The more variance the context has, the less accuracy can be obtained by using a base learner to represent the real model. If base learners are not accurate, the accuracy of the final ensemble of them is surely doubtful.

Thirdly, consideration of the goal of a virtual organization: in our scenario, the main target of each member of the virtual organization is to achieve benefit by sharing its information. Every shop expects that it can get more accurate results if it uses a model based on overall data rather than one based only on its own data. Each shop also understands that there is context heterogeneity. The difficulty is that the shops do not know whether the heterogeneity is big enough to decrease the accuracy of the final model. If it is, then they cannot get the benefit of sharing information.

Fourthly, features of context heterogeneity: in practice, most of the sources of context heterogeneity are immeasurable or unobservable, and available data sets usually do not record them. As in our scenario, there is no one attribute to indicate the source or measure the difference. What we know is that different shops have different catalogs, sell different products and adopt different policies. The context heterogeneity is caused by the combined effect of all these known and any other possible unknown issues. It is also difficult for us to measure the known issues together or separately.

Based on the above discussion, it is our view that existing meta-learning based DDM techniques are not suitable for our scenario. We need a new technique that can deal with unobservable and immeasurable context heterogeneity explicitly in distributed environment. We term this context-based distributed data mining.

III. Our Approach

To solve our problem, we need an approach that can explicitly address context heterogeneity.

A. Towards Context-based Distributed Data Mining

In statistical meta-analysis, a popular way to model unobservable or immeasurable context heterogeneity is to assume that the heterogeneity across different sites is random. In other words, context heterogeneity derives from essentially random differences among sites whose sources cannot be identified or are unobservable [11]. Distributed data across various sites having randomly distributed context heterogeneity is often regarded as conditional IID [13].

The above analysis leads to a hierarchical model which describes context heterogeneity with mixture models and employs latent variables in a hierarchical structure [13, 14].

Assuming that the distribution of context heterogeneity is P_1 with mean μ and variance τ^2, data y_{ik} at k th site has distribution P_2 with mean μ_k and variance σ^2, then the hierarchical model of distributed data is:

$$\begin{aligned} &\text{Between contexts level}: \mu_k \overset{IID}{\sim} P_1(\mu, \tau^2) \\ &\text{Within context level}: (y_{ik} \mid \mu_k) \overset{IID}{\sim} P_2(\mu_k, \sigma^2) \quad k = 1,..., K \end{aligned} \tag{2}$$

When $\tau^2 = 0$, (2) will be the same as (1). So a distributed problem with negligible context heterogeneity is a special case of the generic situation.

In our scenario, we use the most common and simplest format of (2) to model how much money a consumer spends on average at every purchase:

$$\begin{aligned} &\text{Between shops level}: \mu_k \overset{IID}{\sim} N(\mu, \tau^2) \\ &\text{Within shop level}: (y_{ik} \mid \mu_k) \overset{IID}{\sim} N(\mu_k, \sigma^2) \quad k = 1,..., K \end{aligned} \tag{3}$$

In (3) $N(.)$ means normal distribution. If t_k is the random sample error of context level and e_{ik} is the random sample error within k th site, (3) can be rewritten as:

$$\begin{aligned} &y_{ik} = \mu + t_k + e_{ik} = x_{ik}\beta + t_k + e_{ik} \\ &t_k \sim N(0, \tau^2) \\ &e_{ik} \sim N(0, \sigma^2) \end{aligned} \tag{4}$$

B. DISTRIBUTED HIERARCHICAL MODELLING

Once the hierarchical model is formulated as (4), the main task of data mining is model fitting, i.e. calculating the parameters $(\ , t^2, s^2)$.

The traditional and most popular hierarchical model fitting is based on maximum likelihood via iterative generalized least squares (IGLS) [15, 16]. IGLS is a sequential refinement procedure based on GLS estimation. For models such as (4), assume residual matrix $\mathrm{E} = \mathrm{E}_1 + \mathrm{E}_2 = \{e_{ik} + t_k\}$ then its variance matrix is $\mathrm{V} = \mathrm{V}_1 + \mathrm{V}_2 = E(\mathrm{E}_1\mathrm{E}_1^T) + E(\mathrm{E}_2\mathrm{E}_2^T)$. If V were known, and X is matrix of x_{ik}, Y is matrix of y_{ik}, then could be estimated by GLS:

$$= (\mathrm{X}^T\mathrm{V}^{-1}\mathrm{X})^{-1}(\mathrm{X}^T\mathrm{V}^{-1}\mathrm{Y}) \tag{5}$$

If were known, the estimation of random effects matrix * could be obtained by GLS:

$$^* = \begin{matrix} t^2 \\ s^2 \end{matrix} = (\mathrm{Z}^{*T}\mathrm{V}^{*-1}\mathrm{Z}^*)^{-1}(\mathrm{Z}^{*T}\mathrm{V}^{*-1}\mathrm{Y}^*) \tag{6}$$

In (6), Z^* is the design matrix for random effects parameters, *vec(.)* is the vector operator of matrix, and $\mathrm{Y}^*, \mathrm{V}^{*-1}$ can be obtained from:

$$\begin{aligned} \mathrm{Y}^* &= vec(\tilde{\mathrm{Y}}\tilde{\mathrm{Y}}^T) = vec\{(\mathrm{Y} - \mathrm{x}\ )(\mathrm{Y} - \mathrm{X}\ )^T\} \\ \mathrm{V}^{*-1} &= \mathrm{V}^{-1} \tilde{} \ \mathrm{V}^{-1} \end{aligned} \tag{7}$$

Staring with an initial estimate of the fixed effects from ordinary least squares, IGLS iterates between (5) and (6) to convergence, which is judged to occur when two successive sets of estimates differ by no more than a given tolerance (on a component-by-component basis).

According to the block diagonal feature of the variance matrix structure [16, 17], we have:

$$\begin{aligned} &= (\mathrm{X}^T\mathrm{V}^{-1}\mathrm{X})^{-1}(\mathrm{X}^T\mathrm{V}^{-1}\mathrm{Y}) \\ \mathrm{X}^T\mathrm{V}^{-1}\mathrm{X} &= \sum_{k=1}^{K} \mathrm{X}_k^T\mathrm{V}_k^{-1}\mathrm{X} \\ \mathrm{X}^T\mathrm{V}^{-1}\mathrm{Y} &= \sum_{k=1}^{K} \mathrm{X}_k^T\mathrm{V}_k^{-1}\mathrm{Y}_k \end{aligned} \tag{8}$$

$$\begin{aligned} ^* &= (\mathrm{Z}^{*T}\mathrm{V}^{*-1}\mathrm{Z}^*)^{-1}(\mathrm{Z}^{*T}\mathrm{V}^{*-1}\mathrm{Y}^*) \\ \mathrm{Z}^{*T}\mathrm{V}^{*-1}\mathrm{Z}^* &= \sum_{k=1}^{K} \mathrm{Z}_k^{*T}\mathrm{V}_k^{*-1}\mathrm{Z}_k^* \\ \mathrm{Z}^{*T}\mathrm{V}^{*-1}\mathrm{Y}^* &= \sum_{k=1}^{K} \mathrm{Z}_k^{*T}\mathrm{V}_k^{*-1}\mathrm{Y}_k^* \end{aligned} \tag{9}$$

In (8) and (9), K is the total number of sites (shops), X_k is the matrix of $(\mathrm{x}_{ik} \mid k)$, and so on for the other variables. Thus each site contributes its component to the total matrix. In this case, no detailed X, Y needed to be transferred. Only the values of the format $\mathrm{A}_k^T\mathrm{B}_k^{-1}\mathrm{C}_k$ are required to be shared. This coincides with the constraint that detailed data are not allowed to be exposed.

Thus the whole algorithm can be divided into a central subtask:

$$\begin{aligned} \mathrm{L} &= \sum_{k=1}^{K} \mathrm{L}_k \\ \mathrm{M} &= \mathrm{L}_1^{-1}\mathrm{L}_2 \end{aligned} \tag{10}$$

and local subtask:

$$\mathrm{L}_k = \mathrm{A}_k^T\mathrm{B}_k^{-1}\mathrm{C}_k \tag{11}$$

Fig. 1 shows the flowchart of our distributed IGLS.

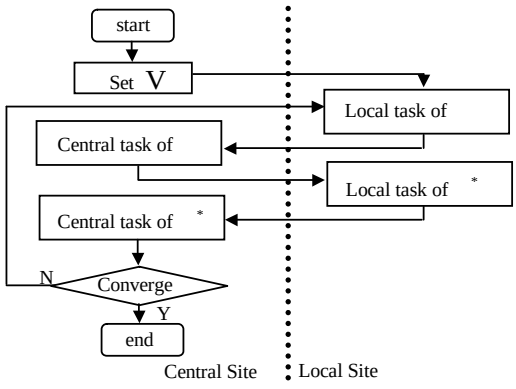

Fig. 1. Flowchart of distributed IGLS

Therefore, our approach has the following three key attributes which are required for a successful DDM algorithm [4]:

- It can be applied in a parallel manner over a group of distributed data sets.
- It provides an architecture to combine local model components.
- It minimizes data transfer.

IV. SYSTEM ARCHITECTURE & IMPLEMENTATION

In order to prove that our approach is feasible in distributed environment such as virtual organizations, a hybrid DDM system has been designed and implemented, which combines both client/server (CS) and agent-based architectural models.

A. Overall Architecture

Fig. 2 shows the architecture of our hybrid system which is

built with the ZEUS Toolkit [18]. The whole system consists of one central site (organization site) and several local sites (sites of individual shops). Detailed consumer and transaction data are stored in DBMS (MS Access) of local sites. Each local site has at least one agent that is a member of the organization. The connection between a local agent and its local DBMS is through Java ODBC. There are three agents at the central site: the agent name server (ANS), the Manger and the Miner.

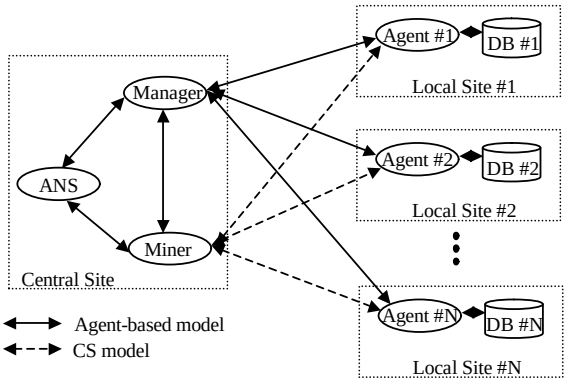

Fig. 2. Architecture of hybrid distributed data mining system

The basic model of our system is a multi-agent system. Routine communication, registration and task planning are done by local agents, the ANS and the Manager. DDM tasks are done through the CS model by the Miner (mining server) and local agents (mining clients).

Our hybrid structure adopts the advantages of both the CS model and agent-based model [3]. There is a dedicated data mining server in our system, which has the ability to handle resource intensive data mining tasks. With agent technology, our system is being able to address the specific concern of increasing scalability and enhancing performance by moving code such as loading algorithms objects instead of data and thereby reducing the communication overhead.

B. Agents

There are three kinds of agents in our system: utility agents, a central task agent and local task agents.

1) Utility Agent

The ANS and the Manager are utility agents. The ANS serves as the name server of the whole multi-agent system. It provides a directory service (Address Books), in which each directory entry contains the Address information (agent name, host IP address, socket port number for receiving communication) about each agent in the system. The Manager plays the role of a facilitator. It provides a directory service in which each directory entry contains the abilities of each agent in the system. It coordinates cooperation and task planning of the system. A DDM task must be first planned by the Manager before it is started. All utility agents are just used by the agent-based model. They do not play any roles in the CS model.

2) Central Task Agent

The Miner is a central task agent. In the agent-based model, it is a generic task agent located at the central site. Once a DDM task is planned, the Miner will start the task after being informed by the Manager. In the CS model, it is a dedicated data mining server. It hosts computing resources such as various kinds of data mining algorithms and strategies.

3) Local Task Agent

All local agents are local task agents. In the agent-based model, a local agent is a generic task agent located at a local site. It can submit its data mining request to the Manager. It can also response to data mining requests of other local agents through the Manager. In the CS model, each local agent serves as a data mining client. According to the data mining sub-task issued by the Miner, a local agent retrieves its local database, performs calculations and returns its results to the Miner.

C. Ontology

The ontology in our system, shown in Fig. 3, is domain specific and consists of registry, mining task and model vocabularies. Concepts in the registry vocabulary are used for system configuration. “Agents” means an agent logs into the system, while “Exit” means an agent logs out. The mining task vocabulary is used to set up data mining tasks in the system. One task is related with one “Process”. One data mining “Request” and at least one “Reply” for that request are preconditions of setting up one DDM task. In the model vocabulary, “Model” means the final result of one DDM task. It consists of two kinds of “Attribute”: independent and dependent attributes. “Hierarchy” refers to the levels of the model. All the concepts and their attributes are list in Table I.

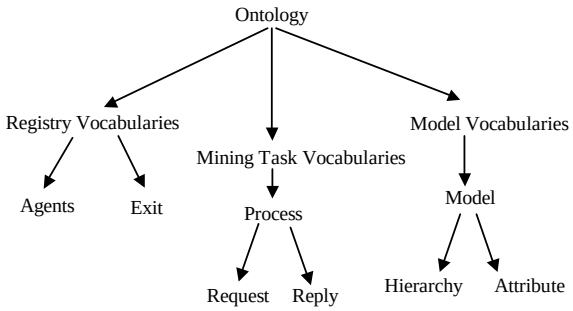

Fig. 3. System ontology

TABLE I
CONCEPTS AND ATTRIBUTES OF ONTOLOGY

Vocabulary	Concept	Attribute
Registry	Agents	Name, host, port, description
	Exit	name
Mining Task	Request	Agentname, model, candidateno
	Reply	Requestid, participant, hierarchy
	Process	Originator, participant, model, hierarchy,status,result,id
Model	Hierarchy	Unit, unitnumber, level
	Attribute	Name, level
	Model	Type, xattribute, yattribute

D. Communication Mechanism

There are two kinds of communication mechanism in our system: messages for the agent-based model and RMI for the CS model. The message mechanism is the inherent communication mechanism of ZEUS agents [18]. With each message communicated as a sequence of ASCII characters, communication between ZEUS agents is via point-to-point TCP/IP sockets. The RMI mechanism is only for the Miner and local agents when they are doing DDM tasks. In our CS model, the Miner is a dedicated data mining server that hosts computing resources. Local agents are data mining clients that host distributed data sets. Through dynamic code loading, the Miner issues subtasks to local agents by transmitting objects that encapsulate data mining algorithms. Local agents download the code of that object, run it and retrieve the local database if required. When local agents finish their data mining subtasks, the results are returned back to the Miner. Thus DDM is realized without transferring local data sets from clients (local agents) to the server (the Miner).

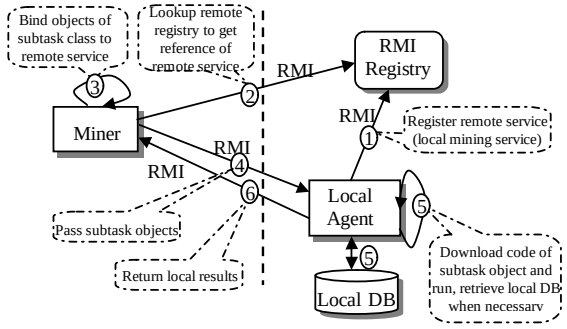

Fig. 4. RMI Communication between the Miner and one local agent

Fig. 4 shows the RMI communication mechanism between the Miner and one local agent. The circled numbers indicate time order. The Miner creates one thread for RMI communication with each local agent. For an example, if there are three local agents to participate in DDM task A, the Miner will start three threads for the RMI communication. All the three threads will terminate when task A is completed. This approach allows the Miner to issue subtasks to more than one local agent simultaneously so that the local subtasks can be executed in parallel. The waiting time that the Miner needs for all local results is

$$T_{Miner\_wait} = \max\{T_k\} \quad (12)$$

T_k : time to excute local subtask by kth local agent

If all local agents reside in different hosts, every one of them will create a RMI registry on each host. If the agents are in the same host, there will be only one RMI registry created.

E. Dynamic Behavior of System

To explain how our system works, three main procedures need to be introduced: system initialization, DDM task planning and distributed data mining.

1) System Initialization

Consider a small agent society consisting of two local agents and the three central agents. The interactions that occur between them at start-up are shown in the interaction diagram of Fig. 5. It shows how the agents (the vertical dashed lines) interact with each other (shown by horizontal arrows). All interactions are achieved through the message passing mechanism.

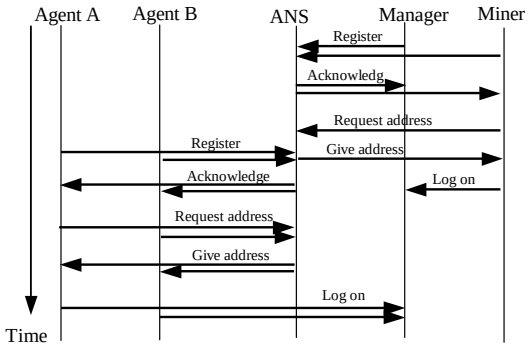

Fig. 5. An interaction diagram of a newly created agent society

The ANS is the first agent to start up. Then the Manager and the Miner start up and get their address from the ANS. After that the Miner logs on to the Manager. So all agents at central site are ready. Local agents register to the ANS one by one and then log on to the Manager. Thus the whole system is started up.

2) DDM Task Planning

DDM task planning is realized by negotiation between the Manager and local agents through the message passing mechanism.

Suppose there are four local agents in the system as in Fig.6. According to its own requirement, Agent A sends a request to the Manager to inform that it would like to do data mining with other agents in the organization. Agent A also needs to give information of model definition (dependent and independent attributes, attribute type (numeric or categorical), model type (linear or nonlinear) with its request. When the Manager receives the request from agent A, it multicasts the request to all other local agents (Agent B, Agent C and Agent D) to ask them if they would like to participate in the DDM task. When the local agents receive the request from the Manager, each of them checks their local database according to the following criterion: is the schema of my database compatible with the model to be built (the same attributes of the same level, the same type)?

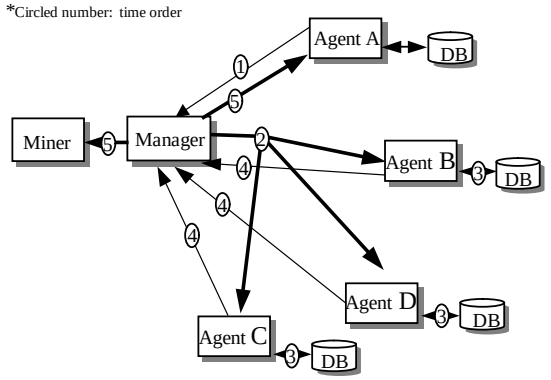

Fig. 6. Negotiation diagram of a DDM task planning

If the answer is YES, the local agent replies to the Manager and adds its name to the participant list of the request (Agent B and Agent C). Otherwise, the agent tells the Manager that it will not participate (Agent D). When the Manager receives replies from all the agents (Agent B, C and D), it sends a message to Agent A to inform it that its request has been approved. At the same time the Manager informs the Miner that a request for a DDM task is approved. Detailed information for the task such as model definition, originator (agent who sends the request) and participants (agents who agree to attend) is also sent to the Miner. The Miner is then responsible for completing the task, while the Manager continues to plan future DDM requests.

This approach allows the Manager to concentrate on organization management and DDM task planning while the Miner focuses on executing tasks. If no other agent participates in a certain task, the Manager informs the originator that no others will attend its DDM task. The task is then done only by its originator and the Miner.

3) Distributed Data Mining

When the Miner receives the message that the DDM task is approved, it begins to execute it. The whole process shown in Fig. 7 consists of three steps: prepare, process and end.

a) Prepare step

At this step, the Miner makes itself, the originator and all participants ready for the DDM task. First the Miner sends messages to the originator and all the participants to ask them get ready for the task. When the originator and the participants receive the Miner's message, each one will check whether any RMI registry exists on its host. If no registry exists, it will create a new registry and register its remote services to the registry. Otherwise it will register its remote services to the existing registry directly. Then each of them sends a message to the Miner to inform that it is ready. After being informed that the originator and all the participants are ready, the Miner starts a thread for the mining task and chooses data mining algorithms according to the model requirements. For each DDM task, the Miner creates a thread for it. The thread terminates when its associated task is finished.

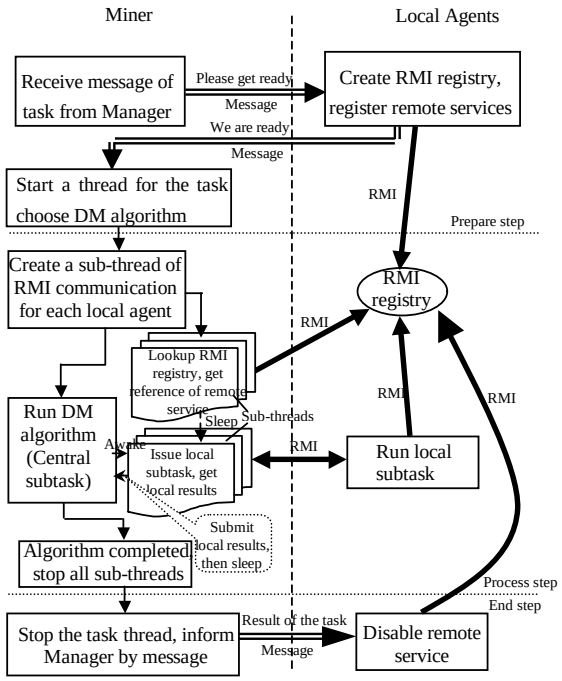

Fig. 7. Process diagram of a distributed data mining task

b) Process step

At this step, the Miner and relating local agents are in the CS model and all interaction between the agents are through the RMI mechanism. After choosing the appropriate DDM algorithm, the Miner starts one sub-thread for each related local agent. That means if there are four local agents for one task, the Miner will creates four sub-threads simultaneously, one for each local agent. Each sub-thread looks up its relating local agent's RMI registry and gets the reference of the relating remote service, then goes to sleep. Meanwhile the main DDM task thread runs the DM algorithm. During the execution of the algorithm, the main thread divides its task related to distributed data sets into several local subtasks and awakes all the sub-threads to let them launch the subtask on their related local agents. When all local results are returned to the main thread, the sub-thread goes back to sleep, and the main thread continues executing its central task. The process of subtask executing is done several times until the algorithm converges. When the algorithm execution is completed, the main thread terminates all its sub-threads. Then the Miner proceeds to the end step.

c) End step

In this step, the Miner stops its main thread of the finished DDM task and sends the mining results to the originator and all the participants of the task. Meanwhile it informs the Manager that the task is finished. All these interaction are done through the message mechanism. When local agents get the mining results, they disable their remote services relating to the finished DDM task.

V. RESULTS

We have applied our context-based DDM approach to data sets that consists of detailed information of consumers from a group of individual shops [19]. Each data set stores detailed data of consumers within one shop. The information includes life-to-date orders, money spent and items bought for each consumer; recency of the first and latest purchases for each consumer; payment methods; very minimal demographics of each consumer. The number (N_k) of consumers within each individual shop (k) is listed in Table II.

TABLE II
CONCEPTS AND ATTRIBUTES OF ONTOLOGY

k	1	2	3	4	5	6	7	8	9
N_k	767	1503	989	2181	1678	1271	3201	2745	2566

We test our approach and system on a distributed regression problem. For each shop, we randomly separate its data into two sets: a training set with 2/3 of the original data and a test set with 1/3 of the original data. Since the distribution of the response variable (average amount of money spent by a consumer per purchase) of every shop is strongly skewed, we normalize the distribution before data mining.

A. Quantity of Context Heterogeneity

The task of model learning is done based on the distributed training data sets. By distributed hierarchical modeling, we obtain a null model, which contains only the response variable and no explanatory variables other than an intercept. The null model provides an initial estimate for the quantity of context heterogeneity as in Table III:

TABLE III
INITIAL ESTIMATION

Iteration	σ^2	τ^2	$\sigma^2 + \tau^2$	$\tau^2/(\sigma^2 + \tau^2)$
6	0.50	0.16	0.66	24%

From Table III, we have the result that 24% of the difference of average money spent is caused by context heterogeneity. From the points of view of statistics and practice, this level of context heterogeneity is medium [11]; it is not low enough for us to neglect. This demonstrates our idea that in our domain of virtual organizations, context heterogeneity does exist across distributed sites and its quantity is not small enough for us to neglect.

Through manually selecting explanatory variables, we get an optimal hierarchical model based on maximum likelihood criterion as shown in Table IV:

TABLE IV
OPTIMAL ESTIMATION

Iteration	σ^2	τ^2	$\sigma^2 + \tau^2$	$\tau^2/(\sigma^2 + \tau^2)$
6	0.24	0.12	0.36	33%

Comparing to the initial estimation, the total variance has been reduced 45%, but the proportion of variance caused by context heterogeneity becomes higher (from 24% to 33%). That highlights the fact that existing variables cannot explain the context heterogeneity sufficiently.

B. Prediction

The task of prediction is done over the distributed test data sets. With the optimal hierarchical model, we obtain the predicted context level residual t_k of k th shop as shown in Fig. 8.

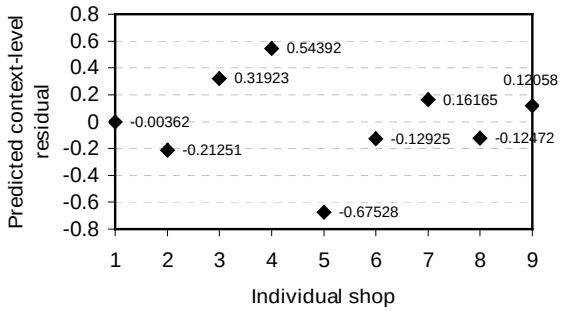

Fig. 8. Predicted context-level residuals of all the shops

From Fig. 8 we know that context heterogeneity among shops 1, 2, 6, 7, 8 and 9 is relatively small. But shops 3, 4 and 5 have relatively high context heterogeneity.

To evaluate the performance of our approach, we compare it with a meta-learning based technique on our data: neglecting the context heterogeneity, the base learners are linear regression models and the ensemble is the simple average of all the base learners. The distributed training data sets and testing data sets are the same as those used in our context-based DDM approach. The comparison is shown in Fig. 9.

From Fig. 9, we can see clearly that our context-based approach has an advantage over the meta-learning based approaches on prediction accuracy, particularly for those shops with higher context-level residuals. This also shows that the limitation of existing meta-learning approaches is most severe when context-level residual is relatively high.

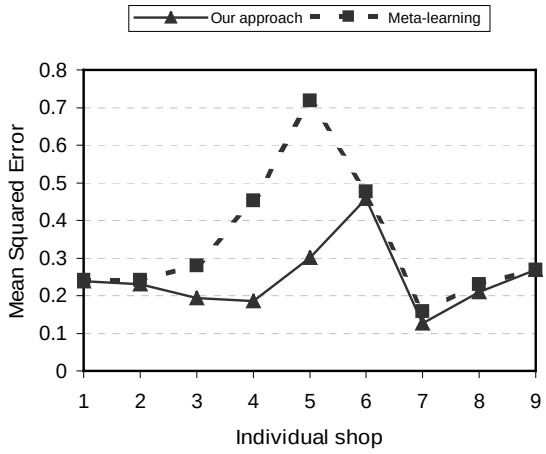

Fig. 9. Accuracy of prediction

Let us consider Shop 6 briefly. Using our context-based DDM approach, its context-level residual is not high but its prediction error is the highest. That means the variance of consumers within this shop is higher than those within any of the other shops. So with our approach, we can determine what the main source of the model variance is (within context heterogeneity or between context heterogeneity).

VI. SUMMARY & FUTURE WORK

Through analysis of the limitations of existing DDM approaches, we have introduced a context-based DDM approach in our application scenario. By encoding context heterogeneity into a hierarchical structure, our approach can address context heterogeneity explicitly when distribution is part of the semantics. To show the feasibility of our approach, we have implemented a multi-agent system to realize hierarchical modeling in distributed environment for regression problem. Our multi-agent system combines CS and agent-based architectural models and employs RMI based and message based communication mechanisms respectively. Our experimental results have demonstrated that our approach is reasonable and feasible.

For future work, we plan to further explore three aspects. Firstly, we aim to extend our approach to classification problems. Secondly, at present, local sites only produces components of the global model, and the iteration is done both on local sites and the central site. Thus the communication grows as required iterations increase. We will try to incorporate a meta-learning framework into our approach. That is, local sites build local models, and these are transferred to the central site to build hierarchical models – we call this proposal context-based meta-learning. In this case, iteration will be done only at the central site, which will decrease communication traffic. Finally, we intend to integrate additional data mining algorithms such as decision trees, neural networks and so on into our proposed framework.

REFERENCES

[1] J. A. Byrne, “The virtual corporation,” Business Week, February 1993.
[2] H. Kargupta and P. K. Chan, "Distributed and parallel data mining: a brief Introduction," in *Advances in Distributed and Parallel Knowledge Discovery*, H. Kargupta and P. K. Chan, Eds.: AAAI/MIT Press, 2000, pp. xv--xxvi.
[3] B. Park and H. Kargupta, “Distributed Data Mining: Algorithms, Systems, and Applications,” to be published in *Data Mining Handbook*, N. Ye, Ed., 2002. Available: http://www.cs.umbc.edu/~hillol/pubs.html
[4] A. L. Prodromidis, P. K. Chan, and S. J. Stolfo, “Meta-Learning in distributed data mining systems: issues and approaches," in *advances in distributed data mining*, K. a. Chan, Ed.: AAAI Press, 1999. Available: http://www.cs.columbia.edu/~sal/
[5] R. Páircéir, S. McClean, and B. Scitney, “Discovery of multi-level rules and exceptions from a distributed database,” presented at Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery & Data Mining KDD2000, Boston, MA, USA, 2000.
[6] K. Tumer and J. Ghosh, “Robust order statistics based ensembles for distributed data mining,” in *Advances in distributed and Parallel Knowledge Discovery*, H. Kargupta and P. K. Chan, Eds.: MIT, 2000, pp. 185-210.
[7] L. Breiman, “Bagging predictors,” *Machine Learning*, vol. 24, pp. 123-140, 1996.
[8] T. G. Dietterich, “Ensemble methods in machine learning,” Lecture Notes in Computer Science, vol. 1857, pp. 1--15, 2000.
[9] G. Valentini and F. Masulli, “Ensembles of learning machines,” in Neural Nets WIRN Vietri-02, Series Lecture Notes in Computer Sciences, M. Marinaro and R. Tagliaferri, Eds.: Springer-Verlag, Heidelberg (Germany), 2002,. Invited Review.
[10] S. Brandt, *Data Analysis: Statistical and Computational Methods for Scientists and Engineers*, Springer, 1998.
[11] M. W. Lipsey and D. B. Wilson, *Practical Meta-Analysis*, SAGE Publications, 2000.
[12] R. Wirth, M. Borth, and J. Hipp, “When distribution is part of the semantics: a new problem class for distributed knowledge discovery,” presented at Workshop "Ubiquitous Data Mining for Mobile and Distributed Environments", 5th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'01), 2001.
[13] D. Draper, “Bayesian hierarchical modeling,” presented at Tutorial on ISBA2000, Crete, Greece, 2000.
[14] I. Kreft and J. D. Leeuw, *Introducing Multilevel Modeling*, Sage Publications, 1998.
[15] H. Goldstein, *Multilevel Statistical Models*, ARNOLD, 1995.
[16] H. Goldstein, “Multilevel mixed linear model analysis using iterative generalized least squares,” Biometrika, vol. 73, pp. 43-56, 1986.
[17] J. M. Bull, G. D. Riley, J. Rasbash, and H. Goldstein, “Parallel implementation of a multilevel modelling package,” Computational Statistics and Data Analysis, vol. 31, pp. 457-474, 1999.
[18] Intelligent Systems Research Group, BT Labs, “The Zeus Agent Building Toolkit,” V1.1 , 1999-2001. Available: http://www.sourceforge.net/projects/zeusagent
[19] The Direct Marketing Educational Foundation, I.N.C., “DMEF Academic Data Sets,” New York, USA.

Yan Xing is a Ph.D. student in the Department of Information Technology, NUI, Galway. She graduated from Huazhong University of Science and Technology, P.R.China with a master degree in pattern recognition and intelligent control. Then she worked as a research assistant in National University of Singapore and as a lecture in Shantou University of China. At present her research concentrates on distributed data mining.

Michael G. Madden has been a lecturer in the Department of Information Technology, National University of Ireland, Galway, since 2000. After graduating with a degree in Mechanical Engineering in 1991, he worked for four years as a research assistant, earning a Ph.D. for his work in developing machine

learning techniques for diagnosis of incipient faults in machinery. He then worked for five years in an internationally trading Irish-based software company, joining as a software engineer and progressing to being the Research & Development manager. His main research interests are Machine Learning algorithms and their application to 'real-world' tasks, particularly in data mining.

Jim Duggan lectures at the Department of Information Technology, NUI, Galway. His research interests include software engineering, operations research and multiagent systems. He has a Ph.D. in industrial engineering and information systems from NUI, Galway. He is a member of the Institute of Engineers of Ireland and the System Dynamics Society.

Gerard Lyons heads the Information Technology Department and is vice-dean of the engineering faculty at NUI, Galway. His interests are in the areas of enterprise computing, distributed multiagent systems, virtual logistics, and biomedical engineering. He holds BA, MS and PhD degrees in engineering and is a fellow of the Institution of Engineers of Ireland. He's a founder of the Information Technology Department's Enterprise Computing Research Group, an advisor to many leading Fortune 100 companies, and a director of a number of consulting and technology companies.