A Mobile Agents System for Intelligent Data Analysis

CLAUDIU IONUT POPIRLAN
University of Craiova
Faculty of Mathematics and Computer Science
A. I. Cuza 13, 200585 Craiova
ROMANIA
popirlan@inf.ucv.ro

LAURA STEFANESCU
Spiru Haret University
Faculty of Financial and Accounting Management
Brazda lui Novac 4, Craiova
ROMANIA
laura.stefanescu@spiruharet.ro

Abstract: The importance of intelligent data analysis in computational vision was a challenging subject and a long motivation for researchers. The main idea of this paper is to develop a mobile agents system for intelligent data analysis (knowledge bases management). In this paper, it is proposed a system which uses a mobile agents architecture. Using this system, sophisticated knowledge bases have been represented and processed without increasing the application size, and with a very good computational time.

Key–Words: Mobile Agents, Knowledge Bases, Intelligent Data Analysis, Java Implementation.

1 Introduction

The research and applications of Artificial Intelligence techniques in data analysis (Intelligent Data Analysis) include (but are not limited to): all areas of data visualization, data pre-processing (fusion, editing, transformation, filtering, sampling), data engineering, database mining techniques, tools and applications, use of domain knowledge in data analysis, evolutionary algorithms, machine learning, neural nets, fuzzy logic, statistical pattern recognition, knowledge filtering, and post-processing. These techniques have been discussed in [5] and [6].

The concept of mobile agent is defined in [1], [12]. They are autonomous objects that can migrate from node to node on behalf of the user who have executed them and they can use the databases or computation resources from clients connected by the network. In order for a mobile agent to be able to migrate, there must be a virtual place, the so-called mobile agents system, which supports mobility. To facilitate effective communication, the agents have to include information about the possible states of knowledge, abilities and preferences of the other agent(s) present in the environment.

Knowledge bases representations has been discussed in [2], and is defined the Knowledge Representation and Processing System. Knowledge representation using interactive network can be found in [7]. The idea of a good knowledge representation and processing [2, 5] is to use different system and mathematical functions to model knowledge. Based on this approach, the semantic schemas was proposed in [8, 9].

In this paper it is proposed a mobile agents system, intended very general and flexible, to effectively improve the data analysis in knowledge bases. For instance, if a certain concept requires a film and a text were a learned, the mobile agent will find the reference to the film on a server and then will search afterwards the prerequisite for concepts the agreement of the film and the text. The mobile agent will bare as a matter of fact subgraph in the graph of knowledge, so that all the entities of proper cognition to is direct his indirect (through descendants) based on perception. It is not adequate for a mobile agent to transport data through network, but it is important to keep information about the location where this data can be downloaded from.

2 The Mobile Agents System Architecture

Using the standard Java support we constructed and manipulated Java classes and objects ([15]) inside a mobile agents architecture to make data analysis within the virtual world. The system architecture that includes the related components is described below and can be visualized in Figure 1.

The role of the view-related components (defined terms) is as follows:

Definition 1 (Agents) The Agents: Local Agent (LA), Presence Mobile Agent(PMA) and Query Mobile Agent(QMA) represents all the agents that contribute to the knowledge base (KB) processing.

Definition 2 (Server) The Server is the computer that implements an local agent, an presence mo-
Definition 3 (Communication Language) The Communication Language \((L)\) is implemented by message passing. An agent that wants to communicate with another agent, first has to create a message object and then send it to the target agent. A message object has a kind and an optional argument object. The receiver agent determines what to do by checking the kind of received message and get parameters as the argument object.

Definition 4 (Root) The Root is a server whose static IP address is known by all the servers in the system. The presence of a knowledge base \((KB)\) on this server is optional.

The system consists of:

- A number \(n\) \((n > 0)\) of Roots;
- Any number of Servers;
- The agents: LA, PMA, QMA;
- The components \((KB1, KB2, \ldots, KBi, \ldots, KBn)\) of a distributed knowledge base.

The mobile agents system will be designed very general and flexible so as not to memorize specific information, but references to them.

2.1 The System User Interface
The user interacts with the system through a Graphical User Interface (GUI), which is implemented on the LA side.

This GUI (Figure 2) is divided into four sections that allow user to:

- configure the LA: choose the way it acts: as a server or as a root, specify the address of the root server and establish how often a PMA will be sent to confirm the presence in the system and to get an updated list of servers.
- specify the local component \((KBi)\) of the distributed knowledge base;
- query the system and view the results.

When set by the user, the knowledge base must consist of entries of the form:

\[ K1 = K2, K3, \ldots, Kn = level \]

where:

- \(K1\) is the knowledge identifier
- \(K2, K3, \ldots, Kn\) - knowledge that extend \(K1\)
- \(level\) - used in a layer-structured knowledge base representation

A well-formed query for the system should consist of comma separated knowledge identifiers. For example: \(K2, K11, K7\). The result of the last query is displayed inside a text area at the bottom of the window. A No data found! result means no knowledge could be found starting from any of the provided knowledge identifiers.

2.2 The System Functionality
The system must be able to process the data from a distributed knowledge base. In this respect, there are Query Mobile Agents (QMAs) which visit, one after another, all or a part of the servers to whom they ask for certain information. When an agent gathers all the knowledge requested by its user it returns home and shows the results. The system must function independently of the server addresses. Thus, in a situation where each server has a dynamic IP address,
the servers list maintained by the agent must also be dynamic. The solution for this list to reflect at any moment, as exactly as possible, the addresses of the servers is to constantly update it. When a server starts, it must announce its presence to be known within the system. At this point, the only address that it knows is the root one. Thus, the server sends to it a PMA with a double role:

- announces the root about its presence (as a server)
- gets from it a list of valid server addresses (optional, to avoid traffic load)

In this way, a new server can see all other existing servers and will also be known to all servers that will come later in the system. The problem is that existing servers cannot see, under any circumstances, the emergence of the new servers. But this is necessary only when you want to create a QMA, which is why, each server will resend a PMA to the root, which may reconfirm its presence in the system and will return an updated list of servers, so that, once it is created and launched, a QMA will start with a list of valid server addresses. Another problem is that a QMA once learned in the process of visitation, has no way to know about the emergence of new servers or about those which were stopped. Thus, a QMA should be able to compare its own list of servers to that maintained by the server and update it if necessary. And here arises a problem because of the way a server updates its list: either when it creates a QMA, or when it is visited by a QMA. Assuming that none of the servers from the list of a QMA doesn’t make any query nor is visited by other mobile agents, this (the QMA) will not be able to update its list of servers and will not know about a possible appearance of a server that could help in its mission. In conclusion, there is one computer (root) which knows all the servers available in the system at any time. Any server that wants to create a QMA firstly obtains from the root a list of servers that are currently in the system and gives that list to the agent as a map. Any server that receives an agent must update its list according to that one from the agent (or vice versa). If a server has no activity during a certain period of time - thus having negative effects on how it succeeds in providing to the agents (QMA) a valid state of the system (list of available servers) - , it will resend a PMA, in other words, it will work on behalf of the system.

2.3 Looking information on servers
The Query Mobile Agent (QMA) migrates through the network having a list of requests (or questions whose answer has to be found on one or more servers it visits) set by the user on behalf of whom it is working. Once the agent gets all the solutions to its problems it can return home.

Let’s imagine a representation of the wishes in the form of $K_x$, where $x$ is a positive number that uniquely identifies a desire (for agent) or a knowledge (for server). An agent member may have as its Wishes List (WL):

$$K_3, K_{18}, K_1, K_{10}$$

A server might respond to the following knowledge, $K_{18}$, with:

$$K_{20}, K_{13}, K_7$$

When the agent queries the server, it will return the following new WL:

$$K_3, K_1, K_{10}, K_{20}, K_7, K_{13}, K_{18}$$

obtained by replacing knowledge $K_{18}$ with new knowledge which will become new desires for agent. These new desires, along with those which don’t yet have an answer will be the query for the next server to visit. As it is natural, understanding of certain knowledge does not imply an understanding of another adjacent, related knowledge. An example of such knowledge is the fundamental geometrical notion of ”point". From here arises the need to define knowledge in primary and complex. Therefore, an agent member may receive from the server, in response to its demands whether new extended knowledge that will be treated as new desires or basic knowledge that will not require further answers. The mission of an agent is considered to be finished when it has only primary knowledge, and its list of desires is empty. An agent (QMA) has no way to know if the server it is going to visit may be useful or not (can provide answers to its requests), and moving on that server could mean just a waste of time and an unnecessary traffic load. There are two ways to decrease the browsing of an agent through the network:

- knowing those servers that have the greatest amount of information, or which gave the most answers to the agents which visited them (best rated servers);
- considering that an agent stores every knowledge that it finds, along with the corresponding address of the server that owns the knowledge, it could communicate those information to every server it visits, and every server could help in this way all the other agents that will come later.
Table 1: Dependent number of servers and computational times

<table>
<thead>
<tr>
<th>Number of Servers</th>
<th>Dimension of List</th>
<th>Transfer Speed</th>
<th>Execution Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.000</td>
<td>4 KB</td>
<td>32 kbps</td>
<td>1 s</td>
</tr>
<tr>
<td>1.000</td>
<td>4 KB</td>
<td>512 kbps</td>
<td>125 ms</td>
</tr>
<tr>
<td>1.000.000</td>
<td>4 MB</td>
<td>512 kbps</td>
<td>62.5 s</td>
</tr>
<tr>
<td>1.000.000</td>
<td>4 MB</td>
<td>1500 kbps</td>
<td>21.3 s</td>
</tr>
</tbody>
</table>

Figure 3: The Java classes of the application

Supposing that each server has an IPv4 address and the port number that it is listening on is always default (does not require memory for storing), it would mean that the list of servers is composed of many entrances to 4 bytes each. As you can see from the Table 1, for a very large number of servers is preferable for a mobile agent to carry only a few of them.

3 The Mobile Agents System Implementation

Regarding the implementation of mobile agents system, we use the object oriented programming (Java language [13]) and Aglets Software Development Kit ([14]). The next classes represent the basic structure of the Java application and can be visualized in Figure 3.

- **KBM.agents**: This package contains classes that will create the agents
- **KBM.tools.SLM**: This package contains classes with role in the administration of the servers list for each of the three types of agents (LA, QMA and PMA)
- **KBM.tools.KLM**: This package contains classes with role in the knowledge list management.

The Java class **KBM.agents.LA** implements the local agent and it can act as a root or as a server depending on its configuration.

The Java class **KBM.agents.PMA** implements the presence mobile agent. The scope of this agent is:

- to announce the current state (available or not) of the server it represents,
- to update the servers list of the server it represents.

The Java class **KBM.agents.QMA** implements the query mobile agent. A **QMA** must know three things:

- the servers list that it will use as a map in the system;
- the user’s wishes list, which it will ask answers for to every visited server;
- a search type: in a knowledge base represented by a semantic schema (layer-structured knowledge base), **QMA** may find and retain all nodes and leaves from the tree with the origin in a set of knowledge given by the user or it can retain the nodes or leaves from a specific level, starting from this set of knowledge.

3.1 Agents transfer through network

The current Aglets implementation uses the Agent Transfer Protocol (ATP), which is an application-level protocol for transmission of mobile agents. ATP is modeled on the HTTP protocol, and can be used to transfer the content of an agent in an agent-system-independent manner. To enable communication between agents, ATP also supports message-passing.

4 Results

The results from the mobile agents system approach for intelligent data analysis in police area (determine the location of drug dealers) and air transportation area is presented. The application represents the components of the distributed knowledge base as Java objects, allowing the mobile agents to make intelligent data analysis and to answer for diverse questions. The important criteria of the application have been considered the easy implementation, functionality and a good running time. A very good computational time has been obtained per each system execution. Sophisticated knowledge bases have been represented and
processed without increasing the file size. The simple scenario (determinate the location of drug dealers) that we will be working with uses a knowledge base structured on three layers:

- level 1: drug dealers
- level 2: cities
- level 3: countries

On the first level there are drug dealers and all the connections between them. A connection from $X$ to $Y$ means that drug dealer $X$ is a supplier for $Y$. Each drug dealer may have one or more links to other dealers and may also have links to one or more places (cities, streets, districts), from the second level, where drugs are delivered. Finally in our example, the third level maintains all the countries where those places belong.

We used for tests a knowledge base Figure 4 composed of 29 entities (13 drug dealers, 11 cities and 5 countries) randomly distributed on 10 servers. The average execution time was 3 seconds per test. During this time a QMA visited each server (one or more times) and then returned home.

For the following results, it is used the next distributed knowledge base (Figure 5) from the air transportation area:

- **Level 1:** Class
  
  $K_1$ = First Class; $K_2$ = Economy Class; $K_3$ = Business Class; $K_4$ = Premium Economy Class;

- **Level 2:** Passenger names
  
  $K_5$ = Allan;...; $K_{56}$ = Vanessa;

- **Level 3:** Fly number
  
  $K_{57}$ = BA5102; $K_{58}$ = BA0208; $K_{59}$ = BA0885;
  
  $K_{60}$ = BA2959; $K_{61}$ = BA0229;

- **Level 4:** Airport name (from)
  
  $K_{62}$ = Cancun; $K_{63}$ = Miami; $K_{64}$ = Bucharest;
  
  $K_{65}$ = Glasgow; $K_{66}$ = Heathrow;

- **Level 5:** Airport name (to)
  
  $K_{67}$ = Miami; $K_{68}$ = Heathrow;
  
  $K_{69}$ = Gatwick; $K_{70}$ = Baltimore;

On each server the user has the possibility to extend the distributed knowledge base through a graphical user interface. The system allows the user to create queries: if the user wants to find out information about the passenger Vanessa ($K_{56}$), the system will process (intelligent data analysis) the knowledge base using mobile agents and will display the result on the interface, as shown in Figure 6. For this test the results ($K_{60}$, $K_2$, $K_{65}$, $K_{69}$) are displayed in 2.2 seconds.

### 5 Conclusions

In this paper an intelligent data analysis approach using mobile agents, object oriented programming
(Java) and Aglets Software Development Kit is proposed. A mobile agents system for knowledge bases processing was developed and tested. The suggested system indicates a method for knowledge bases processing in a distributed environment using mobile agents. There are presented the stages of achieving the system and a running example for a knowledge base in the air transportation area and police area. The results obtained can effectively improve the problem of knowledge bases processing and distributed calculus modeling. As future direction of our research, we shall try the implementation delivered the suggested model and the development of intellective internal model of agent, to inclusion the mechanisms for processing and reasoning.

Acknowledgements: The presented work has been conducted in the context of the GRANT-PCE-II-2008-2-IDEAS-450, *The Management of Knowledge for the Virtual Organization. Possibilities to Support Management using Systems Based on Knowledge* funded by The National University Research Council from Romania, under the contract 954/2009.

References: