WHITE RABBIT – Matchmaking of user profiles based on discussion analysis using intelligent agents

Marc-André Thibodeau, Simon Bélanger, Claude Frasson

Université de Montréal
Département d’Informatique et de Recherche Opérationnelle
C.P. 6128, succursale Centre-ville
Montréal, Canada H3C 3J7

{thibodea,belanger,frasson}@iro.umontreal.ca

Abstract. The White Rabbit system intends to enhance cooperation among a group of people by analyzing their conversation. Each user is assisted by an intelligent agent which establishes a profile of his or her interests. Next, with its autonomous and mobile behavior, the agent will reach the personal agents of other users to be introduced and presented to the ones that seem to have similar interests. A mediator agent is used to facilitate communication among personal agents and to perform clustering on the profiles that they have collected. Conversation between users takes place in a chat environment adapted to the needs of the system.

1 Introduction

In a large business or a large-scale research center, it is extremely common to see significant problems in coordination and cooperation. The resulting mismanagement generally leads to considerable drops in productivity compared to the expected one. The fact is that when a large number of people work in the same organization, everybody is not aware of all the available resources or even of the other projects underway in the organization. Sometimes, different groups of people will work on similar, even identical projects, consequently reinventing the same ideas and concepts twice, or else develop many times the same components instead of combining their efforts and sharing their knowledge. Similarly, resources like experts or past realizations could be used in a profitable way but are not, because of the ignorance of their existence.

This is the problem we are trying to solve. The approach we chose is the use of intelligent agents to discover the similar interests held among a group of people working in a particular domain with the intent to put them in relation and enhance their level of cooperation. The agents analyze conversation between users through a chat interface to build up for each of them a precise profile of their interests. Once those profiles are built, they will be used to categorize users who will finally be introduced to each other when they are classified in a similar category (or cluster).
First, this article presents research advances in the particular domain of intelligent agents for matchmaking by presenting two similar systems. Next, the White Rabbit system’s prototype is explained in detail. It ends analyzing and discussing both the results obtained as well as the future directions that the system may take.

2 Matchmaking Using Agents

This section compares the White Rabbit system with two similar systems developed at the Massachusetts Institute of Technology: Yenta and Butterfly. This discussion highlights White Rabbit’s characteristics.

2.1 Yenta vs. White Rabbit

Yenta is a matchmaking system designed to find people with similar interests and encourage communication between them [1]. This system is composed of a set of decentralized agents that group into categories to represent the user’s similar interests. These categories can then be used to make presentations and enable users to exchange messages with people that are part of the same cluster, i.e. having common interests.

Many agent systems already implemented use a centralized architecture by which an agent serves one or many people at a time. However, such an architecture presents notable disadvantages:

• It is difficult to apply a centralized architecture to large-scale systems. Shardanand and Maes [2] claim that the communication problems which arise through such a system are generally of a quadratic order of complexity.
• A centralized server presents only one location where an accidental breakdown can have severe consequences for a system that has to be reliable and available at all times.

Consequently, future perspectives tend to show that agent systems will use a larger number of agents communicating with one another. This is the approach used by Yenta and the one that we also used for White Rabbit. However, we improved this architecture by adding a mediator agent, which facilitates the communication between the agents, and provides a single rally-point for clustering.

The main idea for Yenta’s clustering consists in an algorithm similar to the hill-climbing algorithm [1]. Once clusters are built, they can be used in different ways. The most important one consists in realizing matchmaking by doing presentations of similar users. This way, the system helps the user to find the expert he or she needs since the expert’s interests, if he or she is represented by an agent of the system, are grouped in the same categories as other users. White Rabbit uses a completely different clustering algorithm, but its use of the clusters found is similar, allowing the user to explicitly ask to be presented to another member of the same cluster and thus facilitating the discovery of an expert.
2.2 Butterfly vs. White Rabbit

Butterfly is another project underway at MIT which presents a lot of similarities with White Rabbit. Butterfly is an intelligent agent system which analyzes conversation over IRC (Internet Relay Chat) in order to suggest to users some discussion channels that may be of interest to them [3]. To do that, it samples the conversations going on into the different discussion groups available and builds, for each user, a profile of his interests. The analysis by agents of real-time discussion is precisely one of White Rabbit’s functions. However, the role of these agents is different. White Rabbit has to form clusters of users with similar interests itself, like Yenta. Instead, Butterfly suggests to users groups that already exist and that have been explicitly created by those users. It has consequently no clustering function to fulfill.

For White Rabbit, we chose to build our own chat system possessing the conversation control capacities and the organization we needed, instead of using the already existing IRC.

The user’s profile of interests used by Butterfly is simply based on a vector of terms associated to positive and negative weights. The actual version of Butterfly uses fixed constants to represent « low », « normal » and « high » levels of interest (e.g. –50, 100, 200). However, more varied weights could be used if the profiles were learned. White Rabbit uses information retrieval techniques to attribute weights to the different concepts that are part of the profile. The way in which the agents learn the user’s profiles in White Rabbit will be described in the next section. One important disadvantage of Butterfly is the fact that it forces the user to explicitly declare his interests to the system and that the profile obtained offers little flexibility. This demands heavy participation of the user who will eventually put the system aside and won’t use it anymore. Automated learning of the profile’s weights solves this problem.

3 The White Rabbit System

Up to now, the project led to the development of a prototype for the discussion analysis agent system. This section presents this prototype’s architecture and explains how its different components work.

3.1 The System’s Architecture

First, we will present White Rabbit system’s global architecture. You can see this architecture in figure 1. Figure 2 shows a personal agent’s architecture in its environment.

The system is made of six principal sections.

- The chat server that organizes the flux of messages through the network;
- A user interface dedicated to both the user and administrator of the system. It allows the user to send and receive messages and to consult and modify his profile.
It allows the administrator to observe and adjust the parameters of the clustering process and to change the knowledge base;

- A personal agent for each user that performs the message content analysis and the presentation service;
- The Voyager layer giving White Rabbit’s agents their mobility and autonomy;
- The PC² knowledge base where different knowledge keywords and links between them are located;
- A mediator agent attached to the server and dedicated to the clustering process and the facilitation of communication between agents.

**Fig. 1.** White Rabbit’s general architecture

### 3.2 User Profile

The whole system works around this essential component. The user profile contains all relevant information on the user’s interests in the chosen domain which will allow agents to discover similarities and consequently to perform an appropriate clustering of users. So it is important that the agent traces a correct portrait of the user it represents.

The approach we adopted for building profiles is based on the PC² knowledge representation developed during the ALICE project, a knowledge extraction system also underway at University of Montreal. This model consists in forming a knowledge (or concepts) graph of keywords to which are linked users via their publications, reports, and projects they are involved in. Concepts are linked to one another by links that have a semantic signification. For example, two concepts can be linked together by similarity links if they are similar or else by specialization links if one is general and “uses” the more specific one. This way, the whole knowledge domain is represented by a graph.
3.3 Learning Module

As we just mentioned, the learning module is the one that modifies the user profile’s weights to make it more and more accurate and realistic. At this moment, this process is made of two steps. We will propose one more during discussion. The first step consists in a preliminary acquisition of information about the user through a questionnaire. The first time he or she uses the system, the user is invited to fill it in and by doing this to give the system keywords reflecting his interests (projects, realizations, expertise). This information is used by White Rabbit to constitute a basic profile for the user. The second step is the one of analyzing discussion. It consists in extracting domain keywords from messages the user sends and then in updating the profile by increasing weights of associated concepts, following a sigmoid function (1)

\[ f(x) = \frac{1}{1 + e^{-x}}. \]

where \( f(x) \) is the new weight and \( x \) is the keyword’s «importance». The keyword’s importance is the number of occurrences of this keyword, modified according to some factors like the similarity degree between concepts, the declarations of interest from the user, etc. The sigmoid function is frequently used in neural networks for computing the values of updates for the link’s weights between units (or neurons). It has the property to vary strictly between 0 and 1 and to have a slow increase for low \( x \)-axis values, then a high increase for medium values and finally slow again for high values on the \( x \)-axis.

Furthermore, since knowledge is represented in a PC\(^2\) form, it becomes possible to update weights of all concepts linked to the keyword discovered by following simple heuristics. For example, all concepts similar to the one found may have their weights modified in the same direction and following the same function, proportionally to the similarity degrees associated to similarity links of the knowledge graph. Consequently, following discovery of a few keywords only, the agent’s learning module can possibly update the weights of all knowledge in the profile. As we will
see, this is very important for our clustering algorithm, Kohonen Maps [4], to work well.

On top of discovering keywords during conversation, the personal agent performs a verification of the profile periodically to allow the user to validate or else invalidate one or more of his profile’s interests. When a keyword’s weight increases over a predetermined threshold, the agent asks the user if he or she is really interested in the associated concept. Weight is then adjusted according to the answer received. This allows the dynamical correction of errors that may have been introduced in the profile during analysis.

So, this second step is done in a totally automatic way, asking only for a minimal user participation who just has to answer to the agent’s periodical questions. This constitutes an important advantage over the Butterfly system presented earlier. Indeed, Butterfly forces the user to explicitly declare his interests and the weights obtained are rigid, permitting the system no flexibility to adapt itself or to correct its errors. In this case, the user must correct himself the errors made by the system by modifying his profile explicitly during discussion. In fact, the user builds his profile himself and Butterfly doesn’t update it in any way. White Rabbit’s discussion interface still allows the user to accelerate learning of his profile by making such explicit declarations, but he or she is in no obligation to do it for his profile to be learned. In the same way, the first learning step (through a questionnaire), is only aimed at simplifying the agent’s task of rapidly building a representative profile of its user’s interests.

3.4 Communication Module

The communication module allows the system’s agents to talk, listen and move to each other. It is entirely based on the ObjectSpace Voyager’s technology [5].

Two of White Rabbit’s aspects demonstrate autonomy and mobility of its agents. First is the discussion analysis of a user by its personal agent. Indeed, after having analyzed and updated its client’s profile, the agent goes to another machine connected to the network to meet one of its colleagues. This one is the mediator agent which must analyze the user profiles and determine in which cluster they are classified and consequently to which group the corresponding users are assigned. The mediator agent gives the personal agent the result of clustering, and finally, the latter returns to its source machine and continues analyzing its client’s discussion.

The second important evidence of mobility happens when the user asks to be presented to a second user who is a member of the same cluster, as determined by the mediator agent. At this moment, the requesting client’s agent (A) will use its autonomy to move to and meet the agent associated to the client to be presented (B). Then, agent A will have the possibility to ask for more information from agent B. If the agent B’s client accepts the request, then, agent B will give to agent A the personal information (real name, email, project description, etc.) on his client that was not set to « private ». A user can set any or all of his personal information to « private » to control what others have access to, maintaining privacy. This is the presentation step following the clustering process.
These two situations demonstrate the strength achieved by the mobility and autonomy of our agents, which allows a considerable reduction of the number of transmitted messages in the network. This way, risks of network overload or lack of resources are greatly reduced, even when the number of personal agents (and equivalently, the number of users) is high.

3.5 Clustering Module

Like we mentioned earlier, the clustering algorithm we used is the Kohonen Map [4] and constitutes an interesting aspect of our system. Indeed, this algorithm invented in 1982 by Teuvo Kohonen, professor at University of Helsinki, Finland, proved for many years to be efficient, being applied to a lot of different domains like medicine [6], physics [7] and seismology [8]. Kohonen Maps are a type of neural network that performs an « unsupervised » learning, i.e. requiring no examples or « good answers » in feedback, contrary to back-propagation neural networks. Since neurons learn in a competitive way, there is no goal to reach nor errors to minimize. So this type of neural network is ideal when output values are not known or hard to measure, as is the case with clustering.

Kohonen Maps are composed of two distinct layers: the input layer and the projection layer (or Kohonen layer). Each neuron present in the input layer represents an attribute, or a dimension of the input data. Each of the input units is connected to all neurons in the projection layer. And finally, each of these connections has an associated weight, generally a number between 0 and 1 so a neuron of the projection layer possesses a vector of weights, each element of this vector corresponding to an input data’s attribute. With this vector, each neuron has the possibility to compute its activation level (or simply activation). The activation is defined by the Euclidean distance given by equation (2).

\[ a = \sqrt{\sum_{i=0}^{n} (\text{weight}_i - \text{input}_i)^2} \]  

We can see that a neuron that possesses a weight vector similar to the activation levels vector of the input nodes will have a low activation level and vice versa. The projection layer’s node having the lowest activation is called the “winner”. In our case, each node of the projection layer corresponds to a cluster. The number of nodes in the projection layer thus determines the number of different clusters in which profiles can be classified. This number can be changed to better suit the number of profiles in the system at a given moment.

But before being used to perform clustering, the network has to be trained. During this training, each input data is presented to the network and activation levels are computed as explained earlier. For each such input, the weight vectors of the winner node and its neighbors are adjusted in a way to approach the input vector. Equation 3 is used to compute the weights variations.

\[ \delta w_i = -\alpha (\text{weight}_i - \text{input}_i) \]  

where $\alpha$ is the learning rate which decreases linearly during the training process and $\delta w$ is a weight adjustment value.

In our system, the Kohonen Map is found in the mediator agent which receives user profiles brought by the personal agents. When a profile is brought, it is first added to a base containing all the profiles. These profiles are then converted one by one into weight vectors that will then be used for activation values of the input units of the neural network. This conversion is trivial, since a user profile makes the correspondence between each keyword in the knowledge base and a real number (the weight), which means that each weight value of the profile can be simply assigned to one input unit of the neural net. The network is trained by making a few passes through the set of profiles in the base. This training leads to the adjustment of the weight values of connections, after which time weights are locked. A cluster is then determined for each profile by computing activation levels and determining the winner for each of them. Finally, the determined clusters are communicated to all personal agents to allow them to update their user’s interface.

Consequently, persons categorized in the same cluster have profiles that are similar and are then listed on each member’s interface (see section 3.6, figure 3, right part). This allows them to greatly reduce the research space when searching for individuals sharing the same interests. And one can easily imagine the time savings it represents in a large company, with hundreds or thousands of employees.

### 3.6 Discussion Environment

We have developed our own discussion environment for the users of White Rabbit instead of using an existing interface like IRC. We wanted to have a discussion environment that allowed us to easily integrate all the discussion analysis and clustering functions we wanted to implement. We also wanted to restrict in some way, through the interface, the discussion in the chosen interest domain. So we have implemented a simple chat environment allowing many people to converse and agents to do their work in an efficient way (figure 3).

The interface allows a user to create and maintain a profile, to visualize it, to discuss and specify, if he or she judges necessary, the type of messages he or she sends (explicit interest declaration, restriction formulations, etc.), to adjust his agent’s activity level, to know the list of users considered having similar interests by the system, to ask for the presentation of one or many of them and to receive and answer questions asked by his personal agent. To insure confidentiality, the user must choose a pseudonym as well as a password to open an existing profile. The profiles and the knowledge base are stored on the server machine.
4 Discussion and Results

White Rabbit is basically a system designed to answer to the needs of a small to medium sized community like the one of a research center, an organization or a company. Consequently, we have not focused on problems related to the system scalability. Instead, we have emphasized the efficiency of the agents work, i.e. by carefully choosing the knowledge representation, by implementing well-known and well-tested learning, clustering and communication algorithms, by making the interface graphical and user-friendly and by using state-of-the-art technologies like the Voyager architecture and the most recent Java version. The prototype implemented is portable on all platforms and aims at forming a solid basis for the development of a useful application, easy to transfer to industries as well as to the different scientific communities. It is also this practical aspect that lead us to chose the real-time discussion analysis instead of user email analysis for example. Indeed, this last alternative would have posed serious problems in a real working environment in terms of accessing information and confidentiality [9].

Basic tests have been conducted to verify the prototype’s efficiency. The first conclusion is the major importance of the knowledge base quality in the quality of the user clustering results. In fact, this process’ quality depends directly on the links between concepts of the base that allow agents to build a correct representation of the user’s interests. If the knowledge graph is incomplete and is not representative of reality, the agent’s learning is inefficient and consequently the clustering step loses all sense. To facilitate the construction of the knowledge base, White Rabbit will eventually be integrated with the ALICE system. This system is composed of a graphical editor which easily allows a knowledge graph in PC² form to be built. We are presently working at collecting statistical data relative to the knowledge base’s
size, to the number of users and to other different parameters having an impact on the clustering quality and the user cooperation improvement that should result.

One more function we propose to add to our system in our future works is a third learning step consisting in asking for a feedback to the user about the clustering produced. This way, the user should be able to easily correct the system by giving it an appreciation of its work. The user would have his profile adjusted according to this appreciation without having to determine himself the causes of his bad classification. Possibilities would be to use a back-propagation neural network to automatically adjust profile weights or, more simply, to use heuristics.

5 Conclusion

In brief, White Rabbit tries to evolve in a system that is applicable to real industrial problem solving. It uses well-known and efficient artificial intelligence techniques to reach this goal. The analysis made by its intelligent agents gives the user a maximum of freedom by asking him as little participation as possible. White Rabbit is really an intelligent agent system where agents are autonomous and move across the network using the Voyager agent architecture, contrary to many existing agent systems. We are convinced that intelligent agents have a real potential and we are confident that their use will answer a lot of actual and future needs of industries and researchers.

References