Cooperative Learning in Self-Organizing E-Learner Communities Based on a Multi-Agents Mechanism

Fan Yang[1], Peng Han[1], Ruimin Shen[1], Bernd J. Kraemer[2], and Xinwei Fan[1]

[1] Department of Computer Science and Engineering
Shanghai JiaoTong University
Shanghai, 200030, China
{fyang,phan,rmshen,xwfan}@mail.sjtu.edu.cn
[2] Faculty of Electrical and Information Engineering
FernUniversitaet Hagen, Germany
bernd.kraemer@fernuni-hagen.de

Abstract. Web based learning provides an unprecedented flexibility and convenience to both learners and instructors. However, it also creates a great number of lonely learners. Hence, there is an urgent need for finding an efficient way to help the learners share their learning experiences and exchange their learning materials during the learning process. This paper reports on an attempt to construct a flexible and effective self-organization system that groups similar learners according to their preferences and learning behaviors. We use a multi-agent mechanism to manage and organize learners and learner groups. Furthermore, we present effective award and exchange algorithms, so eventually learners with similar preferences or interests can be clustered into the same community. Experiments based on real learner data have shown that this mechanism can organize learners properly, and has sustainably improved speed and efficiency of searches for peer students owning relevant knowledge resources.

Keywords: Intelligent agents

1 Introduction

Recently, web based learning technology enables many more students to have access to e-learning environments, which provides students and teachers with an unprecedented flexibility and convenience. On the other hand, this development from classroom teaching to blended or e-learning creates too many lonely learners. While a lot of research has been pursued to provide collaborative learning environments for geographically dispersed learner groups [1], web-based lectures allow instructors and learners to share information and ideas with the entire class, supplemented by multimedia resources, electronic mailing lists and digital video links. But this teacher-centered learning mode bears inherent limitations such as learner passiveness or lack

T.D. Gedeon and L.C.C. Fung (Eds.): AI 2003, LNAI 2903, pp. 490-500, 2003.

of interaction. As a result, there is an urgent need for finding an efficient way to help learners share their learning experiences and insights and exchange learning materials during the learning process. An intuitive way to accomplish this objective is to group learners with similar preferences into the same community and help them learn collaboratively.

A web based learning environment usually involves a great number of active members, including information providers, information users, consultants, tutors or administrative staff. Because the learners involved are generally widely spread and often don't know each other personally, a great challenge is how to find suitable and timely information resources in a distributed environment. To cope with this challenge, middle agents are often used in distributed information systems to facilitate services among users [2], such as the InfoSleuth [3] or the Open Agent Middleware (OAM) of IDIoMS [4]. However, in these distributed systems, the relationship between user agents and middle agents is always pre-defined by a human designer [5]. They have, however, difficulties in handling the dynamism inherent in such open environments. To achieve a good performance and a high scalability, the organizational structure of a socio-technical information system should be both self-organizing and adaptive [6].

In this paper, we present a self-organization model that relies on earlier work by Wang Fang [5]. Around this model we have implemented our own self-organization method to cluster learners automatically and quickly. Section 2 briefly introduces our work, including the underlying conceptual framework and the architecture of our prototype system. The detailed design patterns and the group formation algorithm are presented in Section 3. Section 4 describes some experiments we conducted to demonstrate the formation of user communities and evaluate the efficiency of this mechanism. Section 5 concludes this paper and provides an outlook on future research work.

2 Conceptual Framework

This paper describes the construction of an automatic and effective organization of learners and group agents in distributed e-learning systems. We refer to the concept “E-Learner Community” as a group of learners who share common preferences and mutually satisfy each other's requirements in terms of relevant knowledge resources.

We generated a Learner Agent (LA) acting on behalf of a real learner. LA is in charge of restorations and updates of learning resources. It is well-known that the behaviors of learners are very complex. An LA also handles the requests of a learner and seeks corresponding learning sources from the system.

During the learning process, learners will browse online courses, submit questions or assignments and perform exercises. All of these actions represent the learning interest and intent of the learners. Generally, we view all of them as different resource requests. For instance, browsing courses can be looked at as many http request flows of learning content. The submission of questions represents a request for specific subjects, and so on.

To make our conceptual model more precise, we provide a few formal definitions on which we will also rely in the definition of our group formation algorithm presented in Section 3.2.

Let **G** and **L** be disjoint sets of group and learner names with typical elements g and l, respectively, and let **P** and **R** be sets of preferences and resources. For $P \subseteq \mathbf{P}$ and $R \subseteq \mathbf{R}$ the mapping s: $R \rightarrow P$ models the fact that preference $p \in P$ i*s supported by* resource $r \in R$ if $s(r) = p$. By $R_p = \{r \in R \mid s(r) = p\}$ we denote the set of *all resources in R supporting preference p.* Now we can define a learner agent acting on behalf of a learner l by combining the learner name with the learner's preferences and the resources supporting these preferences.

Definition 1. A learner agent is a triple $A_l = (l, P_l, R_l)$ with $P_l \subseteq \mathbf{P}$ and

$$R_l = \bigcup_{p \in P_l} R_p \tag{1}$$

As the community of learners typically becomes pretty large, it would be a performance bottleneck if the LAs would send requests directly to other LAs. To avoid traffic overload and increase the efficiency of searches, we propose another kind of agent, called Group Agent (GA), to serve as the broker for requests from a smaller community of LAs. A GA is responsible for locating providers of resources and managing the association of learners to communities and it can interact with both the local LAs in its management domains and the other GAs.

The GAs are distributed and only manage local communities of learners, they can enhance the whole system's robustness as there is still some probability of component failure dynamically adjusting learners of the provider sends neighboring GAs on real learners. Furthermore, some GAs may take charge of two or more categories of documents and some of them may lose all of their members during the community formation.

A *multi-agent structure* can be modeled by associating LAs and GAs through the mapping $m: L \rightarrow G$, where $m(l)=g$ denotes the fact that learner l is a *member of the group* managed by g. All *LAs managed by g* are then defined by the set:

$$A_g = \{l \in L \mid m(l) = g\} \tag{2}$$

and the set of resources maintained by the community of LAs managed by g is defined by:

$$R_g = \bigcup_{l \in A_g} R_l \tag{3}$$

Therefore, we propose a two-layer multi-agent structure. Figure 1 illustrates a schematic view of it.

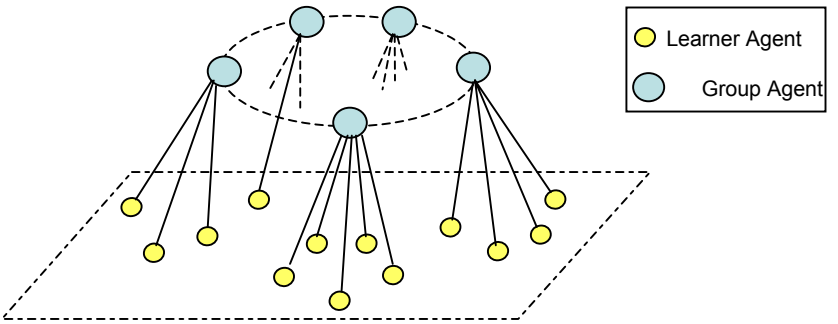

Fig. 1. A schematic view of the two-layer multi-agent structure

The LAs submit resource requests to their GA, and each GA will take care of finding suitable learning resources providers and return the requested resources to the real learner. The interaction between LAs and GAs is transparent to the human learner. Our learner behavior model is based on clustering of learners according to dynamic resource requests and does not require laborious human interference.

3 Core Mechanism and Group Formation Algorithm

We have divided the process of group formation into two main parts:

- Initialization and automatic registration
- Dynamic adjustment and self-organization.

We shall discuss the details of each part in the following subsections.

3.1 Initialization and Automatic Registration

In our open E-Learning platform, preferences and behaviors of learners can be collected from web servers, automatic question–and-answering system, assignment system, examination system, newsgroups or other resources. In order to analyze the behavior of learners, we have to generate reports that cover a large period of time and gather all relevant learning information. Therefore, we have defined the Learner Markup Language, which offers the flexibility to combine all learning information to produce the profiles of learners [7]. As discussed above, we only focus on the preferences and resources attributes. The XML document in Fig.2 shows a typical user profile.

The **users** element is the root element of a User Profile valid document. A **users** element can contain zero or more **user** elements. All **user** elements have global attributes ***ID*** and ***interest***. The ***ID*** attribute is a unique number to identify the **user** element and the ***interest*** are strings denoting the preference of the user. The **resources** element is the container of a **resource** element, which describes the title of a resource owned by a learner object.

```
<?xml version="1.0" encoding="gb2312"?>
<users>
   <user ID="149" interest="Undergraduate Computer Science">
      <resources>
         <resource>Software Develope Methods</resource>
         <resource>Program Design</resource>
         <resource>C Language Program Shortcut</resource>
         <resource>Computer and Information Processing</resource>
         <resource>Data Structure Tutorials</resource>
         <resource>Network Tutorials</resource>
         <resource>C Program Design Instances</resource>
      </resources>
   </user>
   <user ID="148" interest="Computer">
      <resources>
         <resource>Foundation of Computer Application</resource>
         <resource>Program Design</resource>
      </resources>
   </user>
   <user ID="147" interest="Business Administration">
      <resources>
         <resource>Foundation of Probability Theory and Administrative Analysis</resource>
         <resource>Management Behabvior Cases</resource>
      </resources>
   </user>
   <user ID="146" interest="E-Commerce">
      <resources>
         <resource>Customer Relationship Managment</resource>
         <resource>E-Commerce Application Develope Techniques</resource>
         <resource>E-Commerce Application Tutorials</resource>
         <resource>E-enterprise Management</resource>
         <resource>E-Commerce Security</resource>
      </resources>
   </user>
   ......
</users>
```

Fig. 2. Example of a learner profile

In the initialization process, the system employs three procedures:

1. First, an LA is generated for every real learner *l* automatically. Its task is to maintain the profile of *l*. It provides a succinct and valid way to save relevant leaner information.
2. Second, one or more GAs are generated together with an initial multi-agent structure *M* specifying which LAs are managed by which GA. In this step, every GA generates two tables to maintain the information of local learners and other GAs respectively. These tables implement the mappings *s* and *m* introduced in Section 2.
3. We also provide an operation that allows a Learner Agent *l* to de-register from its GA *g* and register with another GA *g′* when required. The effect of this operation is that the pair *(l, g)* is removed from *M* and the pair *(l, g′)* is added to *M*. In order to meet the requirement of frequently changeable relationships between LAs and

GAs, the system maintains a table of all group members and their relationships. Every GA is associated with a mutable table to maintain the information shared by all learners in a local group (we will discuss this in section 3.2). The initialization and interaction of a Learner Agent with a Group Agent is transparent to the real learner.

3.2 Self-Organizing Learner Communities

In our former research, we have constructed a collaborative learning platform based on a multi-agent model [8]. Some investigations have been pursued on the infrastructure and interaction language between multi-agents. In this paper, we focus on automatically grouping learners sharing similar preferences and dynamically adjust learners according to their changeable behaviors.

The main algorithm implementing this search strategy is shown in Figure 3. Besides the preferences and resources of the actual learner community, the algorithm takes variables **Gnum, MaxRequestTime,** and **topSearch** as inputs. They are needed to constrain the number of searches across large communities. Each LA and GA maintains a variable **award**, which is used to maintain information about matching preferences. The local variable "Provider" refers to an LA, while variable "Requester" refers to both LAs and GAs.

INPUT:

1. 1.A **learner.xml** file containing a learner profile for each learner in a community (including the learners' preferences and information about their resources)
2. 2.The number **Gnum** of GAs to be generated
3. 3.The maximum **MaxRequestTime** of request times per learner
4. 4.The maximum **topSearch** of searching times

OUTPUT: A self-organized learner community

PROCEDURE

```
Self-organizing (learner.xml, Gnum, MaxRequestTime, topSearch)
{ Find-Provider();                    //find the resource provider
   if (isSucceed=true) then
   { LocateResource ();               //send the resource to searcher
      Award(requester);
      Award(provider);
   }
   if provider.GA<>requester.GA then
   Exchange(provider, requester);
}
```

Fig. 3. Self-organizing Algorithm

The search schema helps GAs to find suitable learning resources. Here, the requests are titles of learning resources, answers are names of identified providers if

there were any matching resources titles. The main search process **Find-Provider()** can be divided into several steps as follows:

- **Judge the type of requester**
  When receiving an information query message from an agent, the GA will judge the type of the agent. Here, a GA can only communicate with local LAs and other GAs. It can not communicate with other LAs. So a GA can only receive two kinds of requests. One is from another GA, the other is from local LAs.
- **If the type of requester is GA**
  If the requester is another Group Agent, it only searches the local group and delivers the provider information if it exists. Because every GA maintains a table that records the information registered by all of its own LAs, the GA can easily find out whether the required information is owned by one of its LAs. Considering the communication problem, only the GA of a provider sends the confirm message and the GA of a requester only chooses the first returned provider.
  If the type of requester is LA, then the GA will search in a local group first. However, if the information is not available locally, the GA seeks help by forwarding the request to other GAs. These other GAs then check their own databases for a match and deliver any positive feedback to the requesting GA, which passes the feedback directly on to the original requester.
- **If the search does not succeed**
  If no positive answer can be found by interacting with neighbored GAs, the middle agent of the requester stops searching and declares that the search has failed. The **topSearch** here is an upper limit for the number of attempts to prevent endless searches in a large and distributed e-learning environment. Since every LA is registered with a GA randomly in the initialization process, one group may have learners with different preferences. Hence, we adopted Wang's award and exchange schemas [5] aiming at recognizing learner behavior and reorganizing learners accordingly.
- **Award schema**
  When a GA relays search results to a LA, it examines whether the search is successful. If the search is successful, that is, if there is an information provider matching a request, the award of the requester and provider LA are increased by 1. This is because both the requester and provider have made a good contribution to the system.
- **Exchange schema**
  After that, the GA of the requester determines whether the requester and provider are both in its group. If they are not in the same group, the GA of the requester contacts the GA of the provider for a membership exchange such that both the requester and provider - who have similar interests - are registered with the same group. The rule is to move the LA with lower award towards the GA managing the LA with higher award. This hypothesis is based on the belief that a learner with high award usually means that it has either requested or provided useful information to other users. The highly awarded LA is always acting on behalf of the main interest of the group. It is called the

authority learner. Hence an attraction to an authority LA can drive other LAs with similar interests to join the same group quickly.
By means of the award scheme, it is possible for GAs to differentiate between learners who are contributing to their community and those who are passive. This information plays an important role in generating the authority learner and the formation of learner communities. By using the exchange schema, GAs can quickly cluster the LAs with the same preferences, which is essential for a quick decrease of the search time and an increase of the success rate.

4 Experiments

The experiments presented here are based on the real learner from the Network Education College of Shanghai Jiao Tong University. We focus on the evaluation of the usefulness and efficiency of this self-organizing mechanism. For simplicity in the experimental system, we made two assumptions:

- **Hypothesis1**: the resources owned by learners are documents and can be classified into certain categories according to their context.
- **Hypothesis2**: every learner only has one preference or interest and owns zero or more documents of relevant category.

In the experimental setting, we chose 1500 users, and the number of owned documents are 1000 (one document can be owned by one or more learners), the preference category is 10.

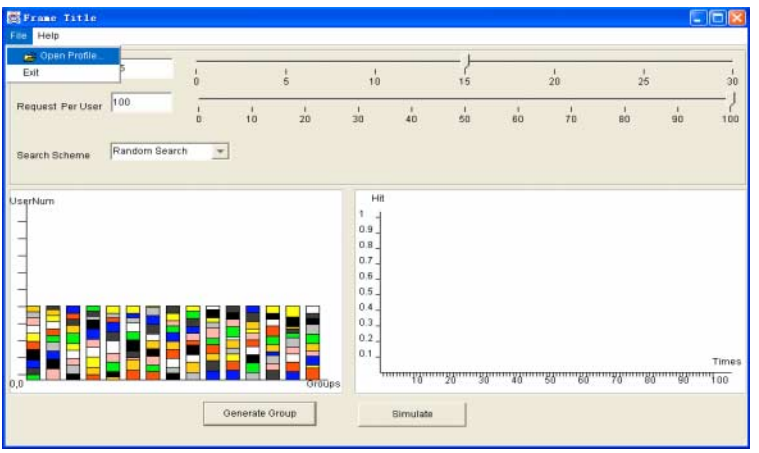

Fig. 4. Introduction of the test platform and system initialization

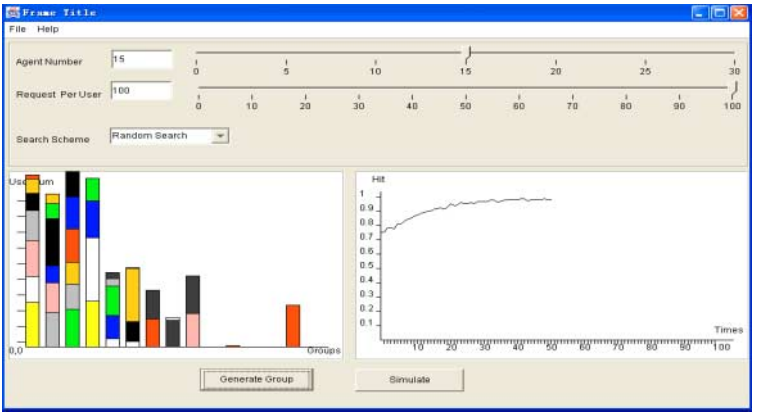

Fig. 5. System situation after 50 requests per learner

Figure 4 illustrates the main test platform of this system. In the top panel, we can define the number of Group Agents (**Gnum**) and the request times per learner (**maxRequestTime**). The left graph at the bottom shows the learner distribution in every group, whilst the right one shows the statistic analysis of the request success rate. When the experiments started, the system generated 1500 Learner Agents on behalf of the real learners and 15 Group Agents. LAs randomly registered with one GA together with the summarized registration information. The GAs kept all information of learners in this group. Figure 4 shows the initial situation in which the colors of every column are mixed and the distribution is almost average.

In the Learner Distribution Graph, every column represents a group, and the colors of the rectangles represent different preferences. The height of each rectangle illustrates the number of learners with special preferences in the corresponding group.

In Figure 5, we can see the situation after 50 requests per learner. We can see the success rate increased quickly from 75% to 91% after 20 requests per learner. In this figure we can see that the trend is toward fewer colors per column and longer rectangles. Also, we can see that some columns become shorter and some even disappeared. That means, some group agents lost all of their users during community formation. This result is consistent with the scenario of this experiment because we only have 10 categories while we generated 15 GA. It is the obvious trend that learners will migrate to the authority GA and some GAs were finally shifted out of the communities, such as GAs 10,11,12,14.

Figure 6 shows the situation after 100 requests per learner. The formation of learner communities was quite successful. It is settled to a stable state with learners who are interested in the same category preferences are all clustered into the same group.

From the RequestSuccess-Rate Graph, we can see that when users are not well organized, the success rate is low. This is because Group Agents often can not find available resources locally and need to send requests to other Group Agents. Since we limited the number of searched GAs, the success rate is lower during the first ten requests per learner. Once learner communities have started forming, however, the system exhibits an obviously improved success rate and greater efficiency. Because

learners who have matching requests and results are gradually grouped together, GAs can more easily find correct answers in their own groups. As a result there is an increase in the success rate of search results. Figures 4 – 6 show that the system's ability to find correct answers to requests was obviously improving, as more and more requests were initiated in the system. And the success rate approached 1 after the learner communities were set up. Meanwhile, the average search time for a request was greatly decreased.

The experiments have been executed using varying numbers of learners and different kinds of learners. The formation of learner communities was always quite successful and can be scaled with the increased number of learners quite effectively.

5 Conclusion

This paper has described a multi-agent environment to self-organize learner communities according to users' preferences and capabilities. Furthermore, we presented effective award and exchange algorithms whose effect is that eventually learners with similar preferences or interests can be clustered into the same community. We evaluated this system in the real e-learning environment operational at Jiao Tong University. The experimental results illustrate that this method can cluster the learners quickly and facilitate appropriate organization between learner agents and group agents. In particular, this method achieves higher search success rates and a sharp decrease of search time.

Our further work will be devoted to extend the adjustment mechanism to handle multiple preferences since learners usually have multiple interests and information resources. Furthermore, we will consider more complex behaviors than mere resource requests. We also plan to run an evaluation with students using the system to verify whether our measurements correlate with the students' satisfaction.

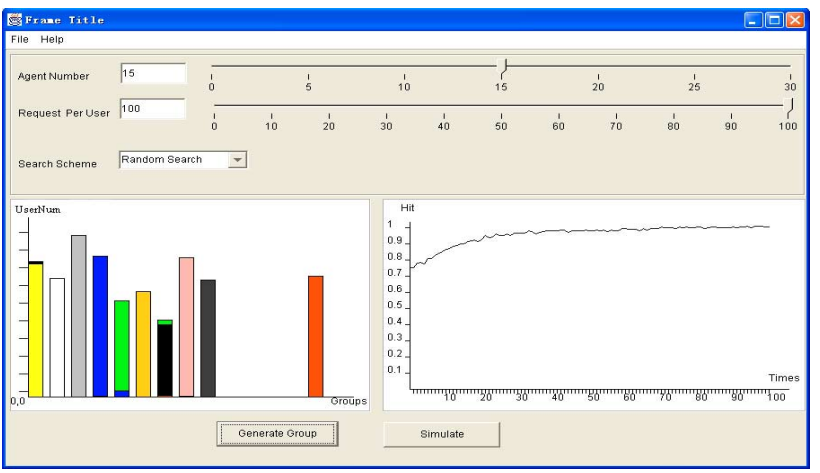

Fig. 6. System situation after 100 requests per learner

References

[1] Suthers, D. Collaborative Representations: Supporting Face-to-Face and Online Knowledge Building Discourse. In: Proceedings of the 34th Hawaii International Conference on the System Sciences (HICSS-34), January 3-6, 2001, Maui, Hawaii.

[2] K. Decker, K. Sycara and M. Williamson, Middle-agents for the internet. IN: Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, Japan, 1997, pp.578-583.

[3] R.J. Bayardo Jr., W. Bohrer, R. Brice, A. Cichocki, J. Fowler, A. Helal, V. Kashyap, T. Ksiezyk, G. Martin, M. Nodine, M. Raschid, M. Rusinkiewicz, R. Shea, C. Unnikrishnan, A. Unruh, and D. Woelk. Infosleuth: agent-based semantic integration of information in open and dynamic environments. In: Proceedings of the ACM International Conference on Management of Data, 1997, pp.195-206.

[4] T. Mohri and Y. Takada, Virtual Integration of Distributed Database by Multiple Agents. IN: Proceedings of the First International Conference on Discovery Science, pp. 413-414, 1998.

[5] F.Wang, Self-organising communities formed by middle agents. In: Proceedings of the First International Conference on Autonomous Agents and Multi-Agent Systems, Bologna, Italy, July 2002, pp.1333-1339.

[6] P.J. Turner and N.R. Jennings, Improving the scalability of multi-agent systems, IN: Proceedings of the First International Workshop on Infrastructure for Scalable Multi-Agent Systems, 2000, pp.246-262.

[7] Ruimin Shen, Peng Han, Fan Yang, Qiang Yang, Zhexue Huang. An Open Framework for Smart and Personalized Distance Learning. In: Proceedings of International Conference on Web-based Learning ICWL 2002, pp19-30, HongKong, China, Aug 2002.

[8] Dazheng Wang, Ruimin Shen, Liping Shen, Collaborative Learning based on Multi-agent Model. IN: Proceedings of International Conference on Web-based Learning: Remote Learning between Men and Machines ICWL 2002, HongKong, China, Aug 2002.