Journal of Zhejiang University SCIENCE
ISSN 1009-3095
http://www.zju.edu.cn/jzus
E-mail: jzus@zju.edu.cn

A novel self-organizing E-Learner community model with award and exchange mechanisms*

YANG Fan (杨 帆), SHEN Rui-min (申瑞民), HAN Peng (韩 鹏)

(*Department of Computer Science and Engineering*, *Shanghai Jiao Tong University*, *Shanghai 200030*, *China*)

E-mail: {fyang; rmshen; phan}@mail.sjtu.edu.cn

Received Feb. 27, 2003; revision accepted Oct. 3, 2003

Abstract: How to share experience and resources among learners is becoming one of the hottest topics in the field of E-Learning collaborative techniques. An intuitive way to achieve this objective is to group learners which can help each other into the same community and help them learn collaboratively. In this paper, we proposed a novel community self-organization model based on multi-agent mechanism, which can automatically group learners with similar preferences and capabilities. In particular, we proposed award and exchange schemas with evaluation and preference track records to raise the performance of this algorithm. The description of learner capability, the matchmaking process, the definition of evaluation and preference track records, the rules of award and exchange schemas and the self-organization algorithm are all discussed in this paper. Meanwhile, a prototype has been built to verify the validity and efficiency of the algorithm. Experiments based on real learner data showed that this mechanism can organize learner communities properly and efficiently; and that it has sustainable improved efficiency and scalability.

Keywords: Self-organizing, E-Learner community, Award, Exchange, Multi-agent
doi:10.1631/jzus.2004.1343 **Document code:** A **CLC number:** TP391.7

INTRODUCTION

Internet technology has enabled students to have access to a web-based learning environment, which provides students and teachers with unprecedented flexibility and convenience. On the other hand, it also creates many lonely learners, which mean those who cannot share their classmates' learning experience and resources like in traditional classroom-based education. While a lot of research has been pursued to provide collaborative learning environments for geographically dispersed learner groups (Suthers, 2001; Christos *et al*., 2002; Krämer and Schmidt, 2001), such as 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, this teacher-centered learning mode bears inherent limitations such as learner passiveness and lack of interaction. An intuitive way to overcome this is to group learners with similar preferences and capabilities into the same community and help them learn collaboratively.

Due to the distributed and dynamic characteristics of e-learners, multi-agent mechanism has usually been adopted to provide collaborative learning. There will be a user agent for each student and a middle agent for each group of students which search suitable service providers in response to the

* Project (No. 60372078) supported by the National Natural Science Foundation of China

learners' requests. However, in these distributed systems, the relationship between user agents and middle agents is often pre-defined by a human designer (Wang, 2002). So they have difficulties in handling the dynamic characteristics inherent from 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 (Turner and Jennings, 2000).

The method of matchmaking, which is the process of finding suitable service provider should also be considered seriously in order to provide high-quality multi-agent based collaborative E-Learning environment. The earliest matchmaker we are aware of is the ABSI (Agent-Based Software Interoperability) facilitator (Singh, 1993), which is based on the KQML specification and uses the KIF as the content language. After that, a lot of researches were devoted to the mechanism and description language on matchmaking, such as SHADE and COINS systems (Kuokka and Harada, 1995), InfoSleuth system (Bayardo *et al*., 1998), the agent capability description language LARKS (Sycara *et al*., 1999), the HTML-like Service Description Language (SDL) and find_nn algorithm presented in (Subrahmanian *et al*., 2000), and so on. For a more comprehensive survey of matchmaking and brokering, see (Klusch and Sycara, 2001).

In E-Learning environment, the objective of community-organizing is to organize the learners which can help each other into a group. Here, we consider the term 'help each other' as 'similar preferences' or 'reciprocal capabilities'. Since the characteristics and behaviors of real E-Learners are very complex; for instance, learners will browse online courses, submit questions or assignment and perform exercises. Hence, we did some investigations on the learning process, both of the learning log files and the learning behaviors. Then we adopted a method to describe the capability of a learner and find the matchmaking learner agents by referring to the SDL and find_nn algorithm mentioned in (Subrahmanian *et al*., 2000). Then, we modified the method to suit the knowledge concept hierarchy and grouped agents in the self-organizing E-Learner community environment.

In this paper, we formalized the Self-Organizing Community (SOC) model that relies on earlier work by Wang (2002); improved its award and exchange schemas with evaluation and preference track records to raise the performance of this algorithm, which can group learners with similar preferences or capabilities automatically and quickly. Section 2 briefly introduces our work, including the underlying conceptual framework of our prototype system, the formal definition of the multi-agent mechanism, the definition of agent capability, evaluation and preference track records. The detailed design patterns, the group formation algorithms, the rules of award and exchange schemas of the self-organization process are discussed in detail in Section 3. Section 4 describes some experiments we conducted to demonstrate the formation of learner communities and evaluate the efficiency and scalability of this mechanism. Section 5 concludes this paper and provides an outlook on future research work.

CONCEPTUAL FRAMEWORK OF THE SELFORGANIZING E-LEARNER COMMUNITY MODEL

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 points.

We generated a Learner Agent (LA) acting on behalf of a real learner. An LA is in charge of not only the advertisement of capabilities, but also the restorations and updates of learning resources. Besides, an LA also handles the requests of a learner and seeks corresponding learning sources from the system.

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 services and managing the association of learners to communities. Furthermore, it can interact with both the local LAs in its management domains and the other GAs.

To make our conceptual model more precise, we provide a few formal definitions.

Definition 1 An LA l is a 3-tuple (C_l,*CET*, *PT*) with $C_l \subseteq C$. Here, C is set of Capabilities the learner has, *CET* is a Capability Evaluation Table and *PT* is a Preference Table (See next sections).

Definition 2 A GA g maintains three data structures: the Type set T, $\sum_C$-hierarchies and a Local-Agent Table (See next sections).

Let G and L be disjoint sets of group and learner names with typical elements g and l. Now we can define the relationship between the g and the local agents.

Definition 3 An agent-community 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:

$$A_g=\{l \in L | m(l)=g\} \tag{1}$$

Definition of learner agent capability

During the learning process, learners will browse online courses, submit questions or assignments and perform exercises. All of these actions represent the learning interest, intent and capability of the learners. Generally, we view all of them as different requests of learning contents (each content) belonging to different knowledge points. For instance, the preferences can be looked at as many http request flows of learning content. The submission of questions represents a request for specific knowledge points of one subject. The marks of each homework or exercise also show the mastery-degree or capability of the relevant knowledge points, and so on.

Here we propose a method to define the learning capability of a learner. We consider the learning capability name can be described as (type: knowledge point), where the 'type' is the kinds of requests and the 'knowledge point' is the knowledge discussed in the learning content. We will give the formal definition as follows:

Definition 4 $\sum_K$-Node is a finite set $K=\{k_1, k_2, \ldots, k_n\}$ of all possible knowledge points capability learners may learn.

Definition 5 $\sum_K$-Hierarchy is a directed acyclic graph $KH=(V, E)$ such that

1) Each vertex of V is a noun $k_i \in K$;

2) For each edge $e=\{k_i, k_j\}$, there is a $\prec$ relationship between two vertexes, we write $k_j \prec k_i$. That is, the concept level of k_i is more general than that of k_j; with k_j being the child of k_i.

Definition 6 $T=\{t_1, t_2, \ldots, t_m\}$: a finite set of all possible types of requests.

Definition 7 If $t_i \in T$ and $k_i \in K$, call (t_i, k_i) a Capability Term. Let $c_i=(t_i, k_i)$, then the Capability Set $C=\{c_i | c_i \in T \times K\}$.

Definition 8 The set of capabilities maintained by the community of LAs managed by g is defined by:

$$C_g = \bigcup_{l \in A} C_1 \tag{2}$$

Definition 9 The matching function f: $C \times C \rightarrow \{0, 1\}$ with:

$$f(c_i, c_j) = \begin{cases} 1, & \text{if } (c_i, c_j) \text{ is a matching pair} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Here c_i is a pair (t_i, k_i), if $t_i=t_j$ and $k_i=k_j$, then we consider (c_i, c_j) is a matching pair.

Definition of evaluation and preference track records

As discussed above, an LA can advertise its capabilities and preferences when it registered to the GA. But the question is that there are always multiple service provider agents who claiming that they have the same or very similar capabilities to accomplish a task in an application. But it is not totally consistent with the actual performance of

provider agents. So we propose two tables to track the performances and preferences respectively.

One is the Capability Evaluation Table (CET). The evaluation records of CET consist of 3-tuples with the form [*t*, *k*, *evaluation*]. The *t* and *k* parameters in the 3-tuple are the type and knowledge point of the evaluated capability respectively. And the evaluation parameter is satisfactory degree returned by the agent who received the service. Such a representation is easy to process and calculate the evaluation of each kind of capability.

The other one is the Preference Table (PT). The preference records of PT consist of 3-tuples of the form [*t*, *k*, *preference*]. Different from the CET, the third item of the record is the preference award of the learner. The bigger the number, the higher is the preference.

SELF-ORGANIZING E-LEARNER COMMUNITY MODEL

In our former research, we constructed a collaborative learning platform based on a multi-agent model (Wang *et al*., 2002). The infrastructure and interaction language between multi-agents were investigated. In this paper, we focus on the mechanism automatically grouping reciprocal learners together and dynamically adjusting learners according to their changeable behaviors. Here we consider 'reciprocal' as the learners with similar preferences and most helpful capabilities. The main algorithms implementing this search strategy are discussed below.

Community-organizing algorithm

The algorithm takes variables *Requester*, *t*, *k* and *n* as inputs. They are needed to constrain the number of searches across large communities. Each LA and GA maintains CET and PT, which are used to gather information on capability and preferences evaluations during the matchmaking process. MATable is the list of matching agents after the performance of the matching subroutines. The local variable "*Provider*" refers to an LA, while variable "*Requester*" refers to both LAs and GAs.

```
INPUT:
    1. The parameter Requester of Agent seeking for help
    2. The type t of requesting capability
    3. The knowledge point k of requesting capability
    4. The number n of maximum searching times per matching
OUTPUT: Learner Communities

PROCEDURE
Community-organizing (Requester,t,k,n)
/* find the most promising providers and return to the requester */
/* insert the evaluation and preference record to provider.CET and requester.PT respectively */
/* dynamically exchange the learners according to both of the CET and PT */
{
 MATable=∅;
 if (t, k)∉T×K return NoAnswer;
 else
 {
    if (Requester.type=LA) then
    {
        MATable= matchingLocally(t,k,n);
        num=count(MATable);
          if (num<n) then
            MAable=PrioriSelect(matchingGlobally
            (t, k, n–num, MATable))
    }
   else MATable=matchingLocally(t, k, n, MATable);
 }
 if (MATable<>NIL) then
 {
   Rank_promise(MATable);
   Community_organizing();
 }
 else return NoAnswer;
}
```

The search schema helps GAs to find suitable learning resources. Here, the requests are capability-terms learners may have; answers are obtained from the matching agent table (MATable). The

MATable includes not only the information on the matching agent, but also the sum-up evaluation of the matching capability in the CET. The main search process can be divided into several steps as follows:

1. Judge the type of requester

When receiving a request 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 cannot communicate with other LAs. So a GA can only receive two kinds of requests. One is from local LAs, the other is from another GA.

(1) If the type of requester is LA

If the type of requester is LA, then the GA will call the subroutine matchingLocally() to 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. Considering the communication problem, only the GA of provider sends the confirmation message and the GA of requester only chooses the first returned MATables. We call this the Priori Selection (As shown in Section 3.2).

(2) If the type of requester is GA

If the requester is another Group Agent, it only searches the local group and returns the matching agent table (MATable) if it is not NIL. Because every GA maintains the a table that records the information registered by all of its own LAs, the GA can easily find whether the required capability is owned by one of its LAs.

2. Judge if the search is successful

(1) If the search is successful

If the search is successful, it will call Rank_promise(MATable) to rank the matching agent according to the average evaluations, and suggest the most promising Agent.

(2) If the search is not successful

If no positive answer can be found by interacting with neighboring GAs, the GA of the requester stops searching and declares that the search has failed.

In next two sections, we will discuss the core subroutines of the main algorithm.

Searching matchmaking agents algorithm

Given a capability term (t, k) and an integer m, returns the MATable, that is, the set of all agents that have the matching capabilities. It is composed of three main subroutines.

1) CreateTccT(t, k). To extend the capability term (t, k) according to the $\sum_K$-Hierarchy. The returned capability-term set TccT is comprised of the children terms of (t, k) if (t, k) is not the leaf node.

2) sql_agent(t, k). To search the LAs from the Local-Agent Table of a GA and find the LAs that matching the capability pair (t, k) abide by the matchmaking function (see Definition 9). It executes the SQL query as below:

```
SELECT   LAs
FROM     Local-Agent Table
WHERE    type=t AND knowledge point=k
```

3) sql_CET(LA, t, k). To calculate and return the average-evaluation of agent LA with capability (t, k).

```
PROCEDURE:
matchingLocally(t, k, m)
/*find the m agents with capabilities the same or closest to (t, c), and output them with the average evaluations*/
  TccT=CreateTccT(t, k);
  isfinish=false;
  SqlA=sql_agent(t, k);
  SqlEV=sql_CET(SqlA, t, k);
  while ¬isfinish do
  {
    insert(SqlA, SqlEV, MATable);
    n=num (MATable);
    if n≥m then isfinish = true
    else
    {
      (t′, k′)=NEXT(TccT);
      if (t′, k′) = NIL then isfinish=true
      else
      {
      (t, k)=(t′, k′);
      MATable=search_agent_local(t, k, m–n,
                MATable);
      }
    }
```

```
    }//end of while
    return(MATable);
}
```

Award and exchange schemas

Since every LA is registered with a GA randomly in the initialization process, one group may have learners with different preferences and capabilities. Hence, we proposed an evaluation record and dynamic exchange mechanism aiming at recognizing learner behavior and reorganizing learners accordingly.

1. Award Schema

When a GA relays search results to an LA, it examines whether the search is successful. If the search is successful, that is, there is a service provider matching a request, then it will first insert the feedback evaluation record from the requester (*t*, *k*, *evaluation*) into the CET of the provider; and then will add 1 to the preference item of the requester's PT, where the preference record is (*t*, *k*, *preference*).

2. 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 can help each other–are registered with the same group.

The exchange rules are:

1) Calculate the average evaluations of the provider;

2) Calculate the award of preferences of the requester;

3) If this preference award is the highest one in the PT and higher than a threshold (an empirical number, here we considered 6), that is, it is the primary preference of the LA, then move the requester LA towards the GA managing the provider LA.

This hypothesis is based on the belief that a learner with high evaluated capability can usually provide useful information to other learners. On the other hand, the highest preference award almost shows the primary preference of the learner. The highly evaluated 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 preferences to join the same group quickly.

EXPERIMENTAL RESULTS AND EVALUATIONS

The experiments were focused on evaluating the usefulness and efficiency of this self-organizing mechanism. Considering simplicity in the experimental system and the visualization effect, we made two assumptions:

Hypothesis 1 The Request Type Set T = {*resource*} and the resources owned by learners are documents and can be classified into certain categories according to their context.

Hypothesis 2 Let $\sum_K$-*Hierarchy* be the concept map as shown in Fig.1, which is part of the concept map of the Operating System courseware.

We give the illustrations of community self-organizing process with 1500 learners from the Network Education College of Shanghai Jiaotong University (www.nec.sjtu.edu.cn), who are learning the OS courseware.

Fig.2 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 part of the graph at the bottom shows the learner distribution in every group, while the right part shows the statistic analysis

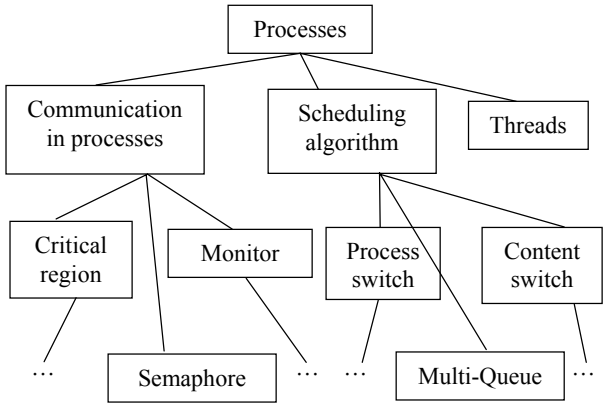

Fig.1 Concept map of $\sum_K$-*Hierarchy*

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. For registration information, learners only provided the keywords of documents they held and their preferences. The GAs kept all information of learners in this group. Fig.2 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 the experiments, we simulated the learners' request actions and generated queries continuously in a certain period. As shown in Fig.3, we can see that after 20 requests per learner, the success rate increased rapidly from 75% to 91%. And some columns became shorter and some even disappeared. That means, some group agents lost all of their users during community formation. It was obvious that learners tended to migrate to the authority GA and that some GAs such as GAs 10, 11, 12, 14, were finally shifted out of the communities. This result showed that the interests of the current 1500 students were focused on ten concept categories, while we generated 15 GAs in the initial time. After 80 requests per learner, the formation of learner communities was quite successful and settled to a stable state with learners interested in the same category preferences are all clustered into the same group.

From the Request Success-Rate Graph, we can see that when users are not well organized, the success rate is low because Group Agents often cannot find available resources locally and need to send requests to other Group Agents. Since we limited the number of searched GAs, the success rate was not satisfactory during the first ten requests per learner. Once learner communities have started forming, the success rate and efficiency increased dramatically; because learners with matching requests and results were gradually grouped together, GAs could find correct answers in their own groups more easily. And the success rate approached 1 after the learner communities were set up. Meanwhile, the average search time per request greatly decreased.

Scalability is an important issue in large information systems. The systems should work properly as more and more users and learning knowledge join the systems. In this work, we mainly investigated the scalability of self-organizing communities associated with increased categories and learning contents. Self-organizing communities have been tested with varied numbers of categories in order to examine their scalability.

Fig.4 shows that the system's ability to find correct answers to queries was obviously improving, as more and more queries were initiated in the system. Fig.5 portrays the continuously decreasing

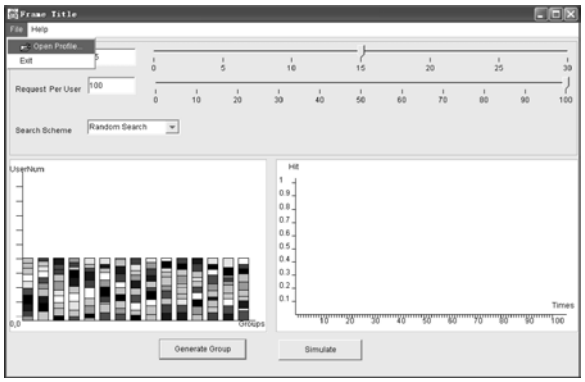

Fig.2 Introduction of the test platform and system initialization

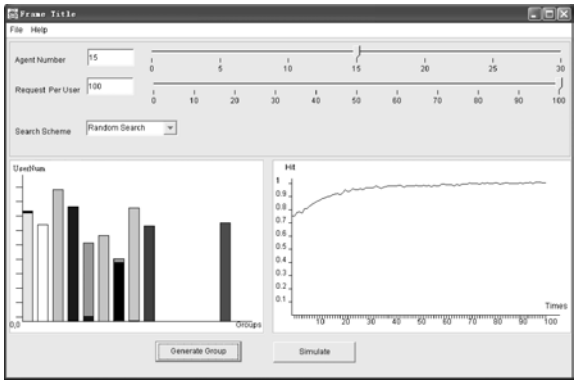

Fig.3 System situation after 100 requests per learner

search time spent on each query. The dot curve in Fig.4 and Fig.5 show respectively the success rate and query time with 1000 categories. The '–' curve shows the situation with 5000 categories. The '–·–' curve shows the situation with 10000 categories.

Fig.4 and Fig.5 show that the user communities were found after every learner launched about 46 queries. With the formation of communities, the actual success rate approached 1. Meanwhile, the average search time for a query was greatly decreased from nearly 600 milliseconds to tens of milliseconds. Due to the communication and processing delay, the query search time oscillated a little after the formation of communities. It did however stabilize to a value of about 20 milliseconds.

The experiments were also constructed 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.

CONCLUSION AND FUTURE WORK

In order to deal with the experience and resources sharing among learners, we proposed a dynamic community self-organization model based on multi-agent mechanism, which can group learners with similar preferences and capabilities. In order to make the search progress more efficient, we introduce the improved award and exchange schemas with evaluation and preference track records to raise the performance of this algorithm. The description of agent capability, the matchmaking process, the definition of evaluation and preference track records, the rules of award and exchange schemas of the self-organization process are all discussed in this paper. Meanwhile, a prototype was built to verify the validity and efficiency of the algorithm. Experiments from real learner data showed that this mechanism could organize learners properly and efficiently; and that the agents' search success rate could be increased dramatically and search time could be much less.

Our further work will be devoted to extending the adjustment mechanism to handle multiple relationships between the GAs and LAs. Furthermore, we will consider more complex characteristics and behaviors of the learners. Meanwhile, we will research how to help the learners learn cooperatively based on the Self-Organizing E-learner Communities.

ACKNOWLEDGEMENTS

The authors are grateful to Professor B.J. Kraemer, Dr. Fang Wang and Professor H.T. Lu for their useful suggestions and comments.

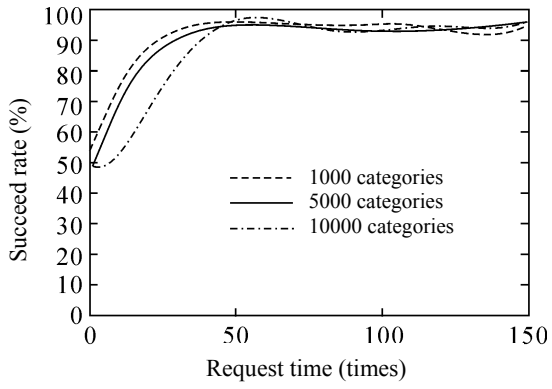

Fig.4 Success rates of query results

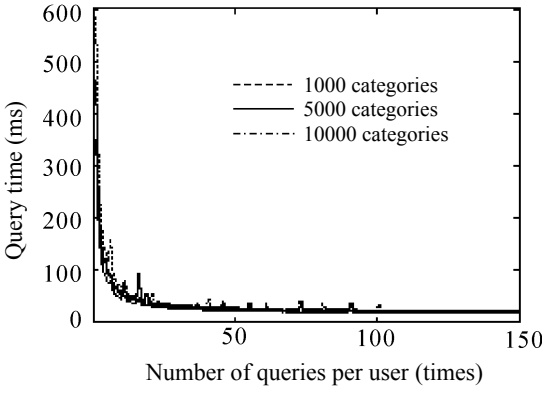

Fig.5 Average query time

References

Bayardo, R.J., Bohrer, W., Brice, R., 1998. Readings in Agents, Michale, N.H., Munindar, P.S. (Eds.). Morgan Kaufmann Publishers, San Francisco, California, p.205-216.

Christos, B., Dimitrios, P., Christos, P., Thrasyvoulos, T., 2002. An Educational Community Using Collaborative Virtual Environments. Proceedings of the First International Conference of Web-based Learning, HongKong, China, p.180-191.

Klusch, B., Sycara, K., 2001. Coordination of Internet Agents: Models, Technologies and Applications, Omicini, A., Zambonelli, F., Klusch, M., Tolksdorf, R. (Eds.). Springer Publishers, Berlin, German, p.56-78.

Krämer, B.J., Schmidt, H.W., 2001. Component and tools for on-line education. *European Journal of Education*, **36**(2):14-41.

Kuokka, D., Harada, L., 1995. Matchmaking for Information Agents. Proceedings of 14th International Joint Conference on Artificial Intelligence. Springer Publishers, Montreal, Quebéc, p.672-678.

Singh, N., 1993. A Common Lisp API and Facilitator for ABSI: Version 2.0.3. Technical Report Logic-93-4, Logic Group, Computer Science Department, Stanford University.

Subrahmanian, V., Bonatti, P., Dix, J., 2000. Heterogeneous Agent Systems. Subrahmanian, V.S., Piero, B., Jurgen, D., Thomas, E., Sarit, K., Fatma, O. (Eds.). The MIT Press, Cambridge, p.151-182.

Suthers, D., 2001. Collaborative Representations: Supporting Face-to-face and Online Knowledge Building Discourse. Proceedings of the 34th Hawaii International Conference on the System Sciences. IEEE Computer Science Press, Maui, Hawaii, p.4016-4026.

Sycara, K., Lu, J., Klusch, M., 1999. Matchmaking among Heterogeneous Agents on the Internet. Proceedings of AAAI Spring Symposium on Intelligent Agents in Cyberspace. AAAI Press, Stanford, USA, p.124-132.

Turner, P.J., Jennings, N.R., 2000. Improving the Scalability of Multi-agent Systems. Proceedings of the First International Workshop on Infrastructure for Scalable Multi-agent Systems. Springer Publishers, Barcelona, Spain, p.246-262.

Wang, D., Shen, R.M., Shen, L.P., 2002. Collaborative Learning Based on Multi-agent Model. Proceedings of International Conference on Web-based Learning: Remote Learning Between Men and Machines. Springer Publishers, Hong Kong, China, p.34-52.

Wang, F., 2002. Self-organizing Communities Formed by Middle Agents. Proceedings of the First International Conference on Autonomous Agents and Multi-agent Systems. ACM Press, Bologna, Italy, p.1333-1339.