퍼지 및 지능시스템학회 논문지 2002, Vol. 12, No. 2, pp. 109-114

온라인 서점 고객을 위한 멀티에이전트 시스템

Multi-Agent System for On-line Bookstore Customers

김종완* · 김상대**
Kim Jong-Wan* and Kim Sang-Dae**

*대구대학교 정보통신공학부
**안동과학대학 마케팅정보학과

요 약

요즘 전자상거래 고객들은 쇼핑몰에 있는 물품들의 가격 정보를 수집하는 비교쇼핑 에이전트들의 도움을 받아서 구매 비용을 절감할 수 있다. 그러나 사용자는 가격이외의 다양한 구매 조건을 만족하는 제품 정보들을 추천하는 에이전트의 개발을 요구하고 있다. 본 논문에서는 에이전트 기반의 전자상거래를 실현하기 위해 다양한 사용자 요구에 적합한 도서 정보를 검색하고 추천하는 멀티에이전트 시스템을 제안한다. 본 멀티에이전트 시스템은 온라인 서점 고객들을 돕기 위해 구현되고 테스트되었다. 실험 결과 전자상거래를 이용하는 구매자에게 여러 온라인 서점의 다양한 도서 판매 조건에 대한 정보를 실시간으로 추천할 수 있게 되었다.

Abstract

Nowadays E-commerce customers can reduce purchasing cost by the help of comparison shopping agents that collect price information of products in the shopping malls. However, user expects a software agent that can recommend product information satisfying various purchase conditions besides price. In this paper, we present a MAS (multi-agent system) which retrieves and recommends book information suitable for various user needs to realize an agent-based E-Commerce. We implemented and tested our MAS to help on line bookstore customers. From the results, we could provide E-commerce customers various book purchase conditions for several online bookstores in real-time.

Key words : multi-agent system, E-Commerce customer, information retrieval and recommendation agent

1. Introduction

With the proliferation of E-Commerce, there are many researches on information retrieval (IR) agents that retrieve and recommend some product information according to user's preference and purchase history [1]. Since an IR agent searches various information on the web and present users the summarized information, users can buy products correspondent to their needs easily within fast time [2].

We need some knowledge base and executing components to implement IR agents according to user need. If we implement the IR agent with the knowledge and functional components in a single agent, the program complexity and the execution overload of the single agent can increase excessively. Thus multi-agent system (MAS) which acts autonomously and cooperates with other agents is needed to solve this information retrieval and recommendation problem in E-Commerce application.

Since the Bargain Finder as a typical IR agent evaluates the value of products in shopping malls with only price information, it does not reflect the exact user need [3]. Because most current IR agents such as the Bargain Finder or the ShopBot do not consider other important conditions about goods like maker and manufactured time, they do not recommend goods correspondent to various user interests [4]. However many E-commerce customers currently want to know the product information with which they can be satisfied. For example, maker, purchase history of specific customer and assessment of others are important to decide buying or not.

In this paper, we present a MAS for book information retrieval and recommendation reflecting user need using the profile which is taken by user. We chose this application to show the usefulness of our MAS for E-Commerce, since there are many on-line bookstores and the bookstores are familiar to us. The proposed MAS presents the information suitable to query offered by user in which publishing year and publisher as well as price is included.

접수일자 : 2001년 11월 1일
완료일자 : 2002년 4월 1일
This research was supported by the Taegu University Research Grant, 2001.

In the next section, MAS architecture for E-Commerce is presented. Section 3 shows the entire processing flow of the proposed MAS. In section 4, implementation and experiments are described. Conclusion is in section 5.

2. MAS architecture for E-Commerce

The proposed MAS is composed of User Agent, Coordination Agent, Information Extraction Agent, and Recommendation Agent. This architecture was inspired from the BIG (resource-Bounded Information Gathering) agent of UMass [5]. BIG is to integrate different artificial intelligence technologies, namely scheduling, planning, text processing, information extraction, and interpretation problem solving, into a single information gathering agent. The proposed MAS and BIG have the same features that they also locate, retrieve, and process information to support a decision process in response to a query. However, differently from BIG, the proposed MAS has no scheduling and planning components but provides a personalized recommendation service. Also, the proposed MAS can control and manage individual agents due to Message Queue component in Coordination Agent. After all, our system is a kind of non-complex information retrieval and recommendation system. As shown in Figure 1, the most important components in the proposed MAS follow in rough order of their invocation. In this section, we mainly describe the facilities of individual agent components in the MAS architecture.

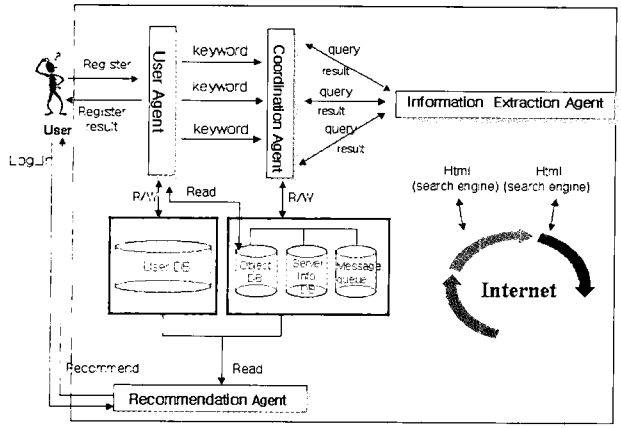


Figure 1. Multi-agent system architecture to retrieve and recommend information

User Agent (UA): In UA, user registers his or her basic data - identifier, password, name, birthday, occupation, sex, and email - and keywords needed to retrieve books. User's preferences about price, publishing year, and publisher are also inputted for each keyword in which user need is reflected. UA provides user the price and the publishing year information in Object DB for the keyword chosen by user in a graphic format. These graphs help user select search criteria of price and publishing year. Then user inputs the specific conditions such as price, publishing year, and publisher referencing the compact graphic information provided by UA.

After UA gets retrieval results using initial retrieval conditions stored as a profile, it continuously learns user's preference with user feedback to the recommendation result of RA (Recommendation Agent). We used two methods to learn user's preference. First, user modifies keyword and preference about keyword directly, or adds a new keyword list. Second, UA automatically updates weights of keywords chosen initially according to user's response.

Coordination Agent (CA): The main objective of CA is to coordinate behaviors between other agents to get satisfactory book information. CA analyzes keywords offered by UA, sends URLs and keywords to IEA (Information Extraction Agent), and requests book information retrieval. After CA receives retrieval results from IEA, it continuously updates Server Info DB, Object DB, and Message Queue.

Server Info DB: The site names and URLs of bookstores and publishers are stored in Server Info DB. They are sent to IEA for retrieving initial book information.

Object DB: There are book names, publishers, ISBNs, publishing years, prices about books retrieved in Object DB. As a new book information is found from IEA, the book information is continuously added to Object DB.

Message Queue: Message Queue maintains keywords used during retrieval process and messages that represent the states of retrieval process. When multiple users request repetitive information retrievals with the same keyword, CA prevents IEA from performing redundant information retrievals.

We used KQML (Knowledge Query and Manipulation Language) format for communication between CA and other agents. KQML represents the most widely used protocol for communication in MAS [6]. In this paper, "insert", "tell", "stream-all" messages are used to process communication protocol of agents. The detailed message processing flow between agents is described in section 3.

Information Extraction Agent (IEA): IEA retrieves HTMLs on the web by using keywords and then stores only book related information into Object DB. Web robots are created by IEA. The number of web robots is same to the number of URLs sent from CA. The URLs taken by web robots are recomposed to fit for individual sites. The retrieved HTML documents are used to filter book name, publisher, ISBN, price and so on. The processed information is sent to CA and then, the contents of Server Info DB and Object DB are continuously updated.

Recommendation Agent (RA): RA recommends users their personalized information with referencing User DB,

Server Info DB and Object DB. RA starts its operation when users login our system. Recommendation process is composed of two steps. In step one, it is performed to identify user and analyze keywords registered by the user. In the next step, RA analyzes three conditions - publisher, price, and publishing year of keywords and finally recommends user books. The recommendation process of RA is described in section 3.4 in detail. The recommendation screen hyperlinks URLs of bookstores with the recommended books.

3. Entire process flow of the proposed MAS

We describe the entire process flow of the proposed MAS based on facilities of four individual agents mentioned in section 2. When user chooses keywords, price, publishing year, and publisher, the proposed MAS starts its operation. Communication between agents is performed by KQML. KQML defines both a message format and a message transmission system that provides a general frame for the communication and cooperation in MAS. In particular, KQML provides a group of protocols for identification, connection establishment, and message exchange. The semantic content of a message is not specified in detail in KQML. Because the standard is open, various languages can be used to exchange knowledge and can be integrated in a KQML message. Figure 2 shows the message exchange process of the UA and the IEA with the principal CA.

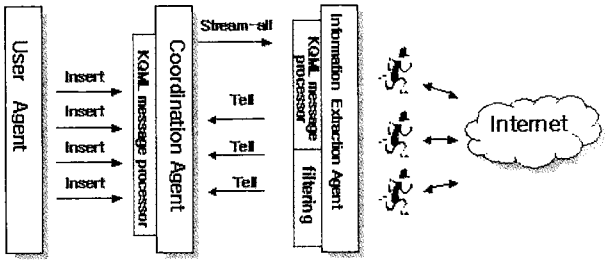


Figure 2. The message exchange process

3.1 Facilities of the UA

UA shows user information in Object DB correspondent to user's keyword in the form of compact histograms as shown in Figure 3. It also stores price, publishing year, and publisher information into User DB, and then transfers the keyword to CA. The message to CA is written in an "insert" performative of KQML. At this time, the message transfer between UA and CA is done by the 1:1 TCP/IP protocol.

3.2 Facilities of the CA

CA identifies the message from UA with KQML message processor and checks it where an error exists. In the case of no error, after CA compares keyword and

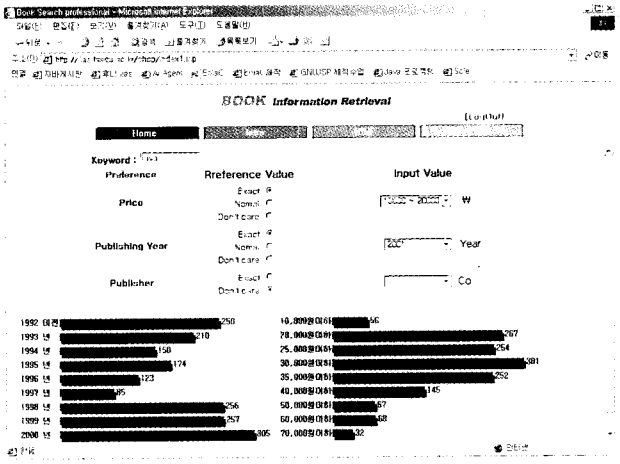


Figure 3. UA processes keyword given by user

time in Message Queue sequentially, it stores and modifies the message. As shown in Table 1, Message Queue is composed of keyword, time, retrieval_state, reply_with, and url_result_num fields. Each field has the following meaning:

Keyword field represents the keyword list transferred from UA. Retrieval_state field has three state values {0, 1, 2}. When retrieval_state equals to 0, it means a new message registration and represents a kind of waiting state to transfer the message to IEA. When retrieval_state has a value 1, it means that CA requests information retrieval to IEA. In the case of 2 retrieval_state, it means that the contents of Object DB are updated with the retrieval results from IEA. Time field maintains time according to variation of state value of each retrieval_state field. Reply_with field is used to check that "stream all" message of CA agrees to "tell" messages from IEA. Finally, url_result_num field maintains the number of "tell" messages transferred from IEA.

Table 1. Message Queue values

key word	time	retrieval_state	reply_with	url_result_num
java	2001-10-23 15:32	1	java_15:32	2
office	2001-10-23 15:45	2	office_15:45	3
linux	2001-10-23 15:49	0	linux_15:49	0
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

We present a message management scheme to communicate individual agents in this paper. The proposed message management scheme is differently performed depending on cooperating individual agents. First, we check the keyword field in Message Queue to communicate between UA and CA. If a keyword transferred from UA is found in Message Queue, CA

operates the following mode: CA modifies the information about the existing keyword when the keyword has been stored within 24 hours. Otherwise, CA deletes the keyword information. On the other hand, if a keyword transferred from UA is not found in Message Queue, CA inserts the keyword as a new keyword into Message Queue. This approach can protect that UA requests CA unnecessary information retrieval.

Second, the communication between CA and IEA depends on the retrieval_state field in Message Queue. The message between CA and IEA is exchanged in the following method:

If CA finds that the retrieval_state of Message Queue has a value 0, then it requests information retrieval to several web robots of IEA with a "stream-all" performative. When the reply_with value equals to the in_reply_to value from IEA, CA maintains current information retrieval status through incrementing the url_result_num value. When any response from IEA is not returned since a primitive time period, CA cancels the old request and tries to demand again IEA with the request message. We can avoid the situation that CA waits indefinitely retrieval results from IEA, as CA maintains the processing time of IEA.

3.3 Facilities of the IEA

After IEA identifies the message from CA with KQML message processor, web robots are created by the IEA. The URLs used by web robots are recomposed to fit for individual sites. Since IEA creates several robots according to the retrieval request of CA and lets the robots search on-line bookstores, we could reduce search time and retrieve information transferred in real-time. The retrieved HTML documents are used to filter necessary book information. IEA sends the filtered information to CA with a "tell" performative.

3.4 Facilities of the RA

RA gives user his or her personalized book information with a ranking system in which keywords, publisher, price, and publishing year given by the user are considered. The recommendation results are continuously changed according to the user preference and then finally RA can recommend users useful information in which user need is reflected.

Recommendation process of RA is as follows:

Step 1 : User chooses the preference for publisher among exact and don't care term.

Step 2 : If user chooses the exact term, RA recommends books with considering price and publishing year term for the publisher given. However, if user chooses the don't care term, RA recommends books according to price and publishing year term for every publisher.

Step 3 : RA finally suggests the books according to the rule set in Table 2. User need is reflected in the rule set.

Table 2 shows the rule set to recommend books according to user's preference. We explain a rule among 9 rules for your reference. As shown in Table 2, if a user selects the exact term as the preference of price and the normal term as the preference of publishing year, then RA recommends the books that match the exact price value and allow some tolerance for the publishing year preference. The other rules are chosen according to the similar criteria.

Table 2. Rule set to recommend books according to user's preference(I: input value, L: low-price, All: all range of pub. year, α: tolerance)

Pub. Year / Price	Don't care	Normal	Exact
Don't care	All L	I$\pm\alpha$ L	I L
Normal	All I$\pm\alpha$	I$\pm\alpha$ I$\pm\alpha$	I I$\pm\alpha$
Exact	All I	I$\pm\alpha$ I	I I

Figure 4 shows a recommendation result of RA using the rule set in Table 2 for a user. The result is obtained when the user chooses a "java" keyword in his or her profile. Also, the proposed MAS supports the user's real-time book shopping since RA directly links book titles in on-line bookstore site.

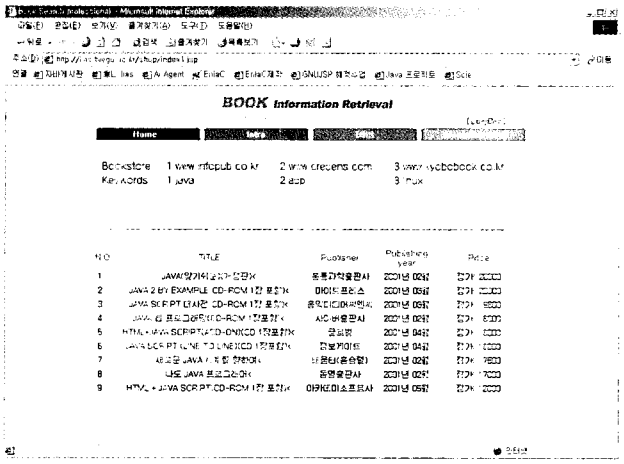


Figure 4. Recommendation results by RA

4. Experiments

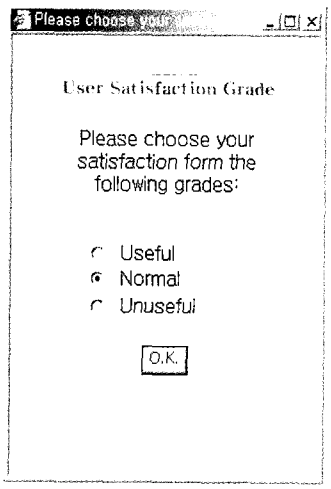
We implemented the proposed MAS for online bookstore customers. The proposed MAS for book information retrieval and recommendation was implemented by using Java, JSP, Servlet, and MS-SQL Server 7.0 under Windows 2000 [7]. Even though we currently implemented the MAS on Windows 2000

server, we could distribute whole system to several servers. Because each agent is implemented in the form of thread and asynchronously operates owing to coordination facilities of CA, it does not need to exist in the same server. We implemented message processors to process messages transferred between individual agents according to KQML protocol in a Java class.

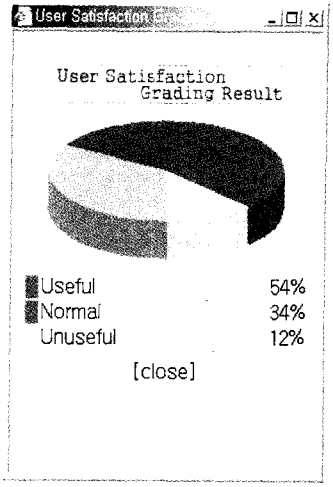
Since CA coordinates IEA by using Message Queue, we can remove redundant information retrievals of IEA and consequently minimize book retrieval time. Thus the proposed MAS can provide E-Commerce customers book information analyzed for on-line bookstores in real time. At this time, not only price information but also extra product information is provided to the customers.

As RA retrieves user's preference every time and constructs recommendation results, user can get a customized information. Resulting from the service of RA, the accessibility of the proposed MAS is improved. According to the recommendation results of RA, user directly feed back to the results by choosing a condition among {useful, normal, unuseful} shown in Figure 5(a). We conducted experiments about degree of user satisfaction with 50 users. From the experimental results are shown in Figure 5(b). As you can see, 54 percentage of users are satisfied at our system and 12 percentage are unsatisfied at that. The rest of users feel the information is helpful to them. Since this experiment is short, we have tested our system continuously to get more reliable results.

We observed that the degree of user satisfaction increased by the following two cases as we used more frequently the system: 1) User directly updates keyword or the weight of keyword, 2) UA continuously trains user's response to the recommendation results of RA. Also, several users replied to unuseful pointed out the shortage of current selected terms such as price, publishing year, and publisher. For example, some of them prefer books of a particular topic.



(a) Users feedback



(b) experimental results

Figure 5. User satisfaction experiments through user feedback

5. Conclusions

In this paper, a system which gives the information about several book shopping malls for on-line bookstore customers in real time is presented. To achieve this goal, we implemented the MAS which was composed of 4 individual agents - user agent (UA), coordination agent (CA), information extraction agent (IEA), and recommendation agent (RA). Currently, the proposed MAS was tested three on-line bookstore sites including Gyobo (http://www.kyobobook.co.kr), Samsungmall (http://www.cresens.com) and Infobook (http://www.infopub.co.kr).

The proposed MAS have several advantages over the existing similar agent-based E-Commerce services. First, the proposed MAS can give users product information in on-line shopping sites in real-time. Second, the proposed MAS can provide users not only price information but also extra product information, and consequently help the purchase behavior of users proactively. Third, an easy graphic user interface is introduced to help users choose purchase conditions. In this interface, the historical data about price and publishing year given by other users are displayed in the graph form.

This research, the application of the BIG architecture to comparison shopping, is a novel combination of existing state-of-the art techniques. So our work might stimulate others to develop promising alternatives. However the proposed system currently provides only a service about book information. In the future, we have a plan to extend other domains required planning or interpretation would inherently be more significant to approach. Further research is needed to improve learning capability of UA and induce association rule from user's purchase transactions.

References

[1] J. Ben Schafer, Joseph Konstan, John Riedl, "Recommender Systems in E-Commerce", Proceedings of the ACM Conference on Electronic Commerce, http://citeseer.nj.nec.com/update/253323, 1999.

[2] P. Maes, "Agents that reduce work and information overload," Communications of the ACM, vol.37, no.7 pp. 31-40, 1994.

[3] Joseph Williams, "Bots and other Internet Beasties", pp. 257-263, 1996.

[4] Robert B. Doorenbos, Oren Etzioni, and Daniel S. Weld, "A Scalable Comparison-Shopping Agent for the World Wide Web", Proceedings of the First International Conference on Autonomous Agent, 1997.

[5] V. Lessor, B. Horling, F. Klassner, A. Raja, T. Wagner, S. XQ. Zhang, " BIG: A Resource-Bounded Information Gathering Agent", UMass Computer Science Technical Report 98-03, 1998.

[6] Tim Finin, Yannis Labrou, James Mayfield, "KQML as an agent communication language", In Jeff Bradshaw (Ed.), Software Agents, MIT Press, http://www.cs.umbc. edu/kqml/papers/, 1997.

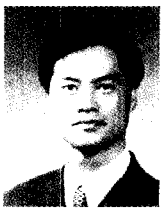
[7] J. P. Bigus and J. Bigus, "Constructing Intelligent Agent with Java", John Wiley & Sons, New York, 1998.

저 자 소 개

김종완 (Jong-Wan Kim)

1987년 : 서울대학교 컴퓨터공학과 졸업 (학사)
1989년 : 서울대학교 대학원 컴퓨터공학과 졸업 (공학석사)
1994년 : 서울대학교 대학원 컴퓨터공학과 졸업 (공학박사)
1995년 - 현재 : 대구대학교 컴퓨터정보공학부 부교수
1999년 - 2000년 : 미국 U. of Massachusetts Post Doc.

관심분야 : 지능형 에이전트, 퍼지시스템, 인공지능, 전자상거래.
E-mail : jwkim@taegu.ac.kr



김상대 (Sang-Dae Kim)

1992년 : 영남대학교 경영학과 졸업 (학사)
1994년 : 영남대학교 대학원 경영학과 졸업 (경영학석사)
2001년 : 영남대학교 대학원 경영학과 재학중 (마케팅)
2001년 - 현재 : 안동과학대학 마케팅정보학과 겸임교수

관심분야 : 에이전트, e-CRM, 전자상거래.
E-mail : osudang@yahoo.co.kr