Available online at www.sciencedirect.com
SCIENCE DIRECT®

Decision Support Systems 37 (2004) 199–213

Decision Support Systems

www.elsevier.com/locate/dsw

Applying case-based reasoning and multi-agent intelligent system to context-aware comparative shopping

Oh Byung Kwon*, Norman Sadeh

ISRI, School of Computer Science, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA

Received 22 May 2002; received in revised form 22 January 2003; accepted 23 January 2003

Available online 14 February 2003

Abstract

Comparative shopping is a promising web service in the field of mobile commerce. This paper aims to propose a context-aware comparative shopping. Multi-agent intelligent architecture is adopted to implement the autonomous negotiation mechanism between buyers and sellers. To automatically estimate user preferences to determine the best purchase, case-based reasoning and negotiation mechanism are utilized. We developed a prototype system and experiment to show the possibility of the mechanism proposed in this paper. We found that our mechanism with multi-agents yields more pay-off, total sales, and wins than the system without those features.

Keywords: Mobile commerce; Comparative shopping; Multi-agent intelligent system; Case-based reasoning; Context-awareness

1. Introduction

The number of users of mobile terminals (phones, PDAs, and communicators) is increasing rapidly. The miniature size of mobile terminals, and that they easily fit into a pocket, makes them an ideal channel for offering personalized and localized services to mobile users. Mobile commerce creates a broad range of new business opportunities for players in the field, such as content and service providers.

One potential business opportunity includes suggesting alternative or comparable products to shoppers, from on- or off-line shops, through mobile devices. In the past, there has not been communication or competition *in real-time* between off-line sellers and on-line sellers. However, if buyers carry their own wireless devices, they can compare products online even when they are shopping at a traditional brick-and-mortar shop. If this kind of ubiquity and "reachabililty" is incorporated, buyers may increase their satisfaction level by making more informed purchases—whether with on- or off-line businesses.

However, only a few web sites are using autonomous agents to negotiate on behalf of their owners, and hence they do not negotiate with buyers. They only provide ads such as product description, price, discount, and warranty condition. There has been substantial research of market-based systems [7,29]. They tried to model an optimization problem for a marketplace consisting of multiple agents in order to calculate optimal equilibrium. Enterprise

* Corresponding author. Tel.: +1-412-268-1304; fax: +1-412-268-1328.
E-mail addresses: obkwon@cs.cmu.edu (O.B. Kwon), sadeh@cs.cmu.edu (N. Sadeh).

0167-9236/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S0167-9236(03)00007-1

[19], Challenger [5], and WALRAS [6] were such systems. However, as many optimization approaches have encountered, purchasing behavior is hard to quantify. Even though it is possible, creating and changing mathematical models are knowledge-intensive—and hence very costly. To increase customer satisfaction, web sites may dynamically vary products' selling conditions by observing customer preferences and behaviors. For example, some buyers may be prioritizing on price, and others may see warranty services as the more important factor. These lead to the motivation to build an intelligent system that can successfully provide a better proposal enough to sell its own products and at the same time yield a profit for itself. Moreover, the service should be fast enough to influence buyers' decisions before they finish shopping at an off-line shop. As a result, the prompt and adaptive mobile web service requires intelligence and autonomy. These naturally lead us to apply intelligent agent-based systems.

MIT's Media Lab has proposed an excellent approach to agent-based negotiation at point of sale. They combined ideas from electronic commerce and mobile environments in agent-based transaction systems [42]. They extended the Kasbah system [4], letting buyers and sellers create their own agents. When PDA-equipped buyers want to make a purchase, they need to know if there are any other shops which are suggesting better conditions. In this situation, the goal of the system would be to successfully find such on-line shops by communicating with several selling agents and comparative shopping agents on the buyer's behalf.

However, the situation needs to be more generalized to be used in a more realistic setting. First, sellers at the point of sale may be extended from one physical marketplace and multiple on-line marketplaces to multiple physical marketplaces and multiple on-line marketplaces. To do so, the location of the buyer at the point of sale and that of the other off-line shops should be considered. The "Impulse" research project is one that enables location-based agent assistance [41]. Secondly, buyers and sellers tend to maximize their own utilities, rather than optimize price level only. Price matters but multi-parameter on-line purchase decision is needed for more sophisticated agent system [23,24]. The negotiation criteria should be augmented from price only to price, quality, brand name, warranty services, etc. Hence, the system must let buyers and sellers create their own profiles. Finally, in comparison to the optimization approach, agents need to be intelligent enough to give sufficiently satisfied suggestions even though utility functions are unknown mathematically.

Hence, the aim of this paper is to propose a context-aware and autonomous system for mobile and comparative shopping that meets the abovementioned requirements. We adopted a multi-agent intelligent system (MAIS) architecture for the following reasons. First, we assume that many selling agents are ready to service according to the request delivered by a negotiator. Secondly, an intelligent agent can contain transaction rules to intelligently and autonomously produce proposals under the delegation of its human owner(s). Next, agent architecture shows widely distributed services very well. Finally, to deal with different buyers' diverse preferences, personalization is needed. Personalization is, to a very limited extent, already available today and an agent system can make it possible. Agent technology has already been used with client/server models and their extensions to build mobile commerce applications [30].

Case-based reasoning (CBR) capability is involved in our prototype since we assume that a user's utility function is hardly represented as a mathematical function. CBR is an AI methodology that provides the foundations of a technology for intelligent systems [15]. The methodology consists of indexing cases, retrieving the best past case from memory, adapting the old solution to conform to the new situation, testing whether the proposed solution is successful, and learning to prohibit solution fails. CBR has been viewed as a technology for automated, intelligent problem solving [37].

We developed a prototype system and experiment to show the possibility of the mechanism proposed in this paper.

The rest of this paper is organized as follows. Section 2 reviews existing research on comparative shopping. In Section 3, we describe our multi-agent framework. The architecture of our purchase advisory system and detail agent behavior algorithm are shown in Section 4. In Section 5, we present a prototype system and experimental analysis to show the feasibility of the idea and we conclude in Section 6.

2. Literature review

One of the definitions of mobile commerce is any type of transaction of an economic value having at least at one end a mobile terminal and thus using the mobile telecommunications network [36]. According to this definition, mobile commerce represents a subset of all e-commerce transactions, both in the business-to-consumer and the business-to-business areas.

Comparative shopping is one of the most plausible web services under mobile commerce, as well as electronic commerce. Comparative shopping services are obtained by integrating several stores, providing the user with a uniform interface for posing requests, and having the application interact with the different stores to find the best bargains [13]. CompareNet [8] and Dealpot [9] are some instances. From the customer's viewpoint, the provision of electronic stores makes comparative shopping possible, allowing customers to browse, compare, and order goods selectively.

The issues for comparative shopping at the point of sale encompass:

- Infrastructures for enabling comparative shopping
- Web service discovery
- Service comparison and negotiation

First, Short Message Service (SMS), Unstructured Supplementary Services Data (USSD), Cell Broadcast (CB), SIM Application Toolkit (SAT), Wireless Application Protocol (WAP), Web Clipping, and Mobile Station Application Execution Environment (MexE) represent enabling technologies for mobile commerce, including comparative shopping. Global Positioning System (GPS) is a system that consists of 24 satellites that orbit in a particular constellation to each other so that several satellites fall within line of sight for any GPS receiver. Cell of Origin (COO) can be used as a location-fixing scheme for existing customers of network operators, but it is not as exact as the other methods.

Next, to find web services in a more efficient and intelligent fashion, Semantic Web technology is emerging. The realization of the Semantic Web is underway with the development of new AI-inspired content markup languages, such as OIL, DAML + OIL (http://www.daml.org/2000/10/daml-oil), and DAML-L (the last two are members of the DARPA Agent Markup Language (DAML) family of languages) [21]. Using DAML markup, one can provide a declarative advertisement of service properties and capabilities which is computer readable.

Finally, several comparative shopping tools, based on technology such as Jango [12] or Junglee [28], have already been introduced and are in widespread use. These tools work on servers connected to a central product database, or on an infomediary such as a portal. They generally assume that the data source can be easily accessed and that data is delivered rapidly and reliably [27].

An emerging technology to find web resources that offer a specific services is "meta-services": a program that provides the user with an interface to perform comparative shopping. Given a request from the user, the program accesses several such services in parallel providing each of them with the request. It then processes the information obtained from the services and presents it to the user. Examples of the meta-services are MetaCrawler [32] and Savvy Search [31] for search engines, and BargainFinder Agent [1] for comparative shopping.

Similar techniques have been used in comparative shopping agents to extract information from specific sites of on-line stores [11]. SmartClient is a distributed agent-based architecture for gathering information. It implements navigational features that can be tailored to the exact needs of each user. It offers solutions to capture the initial large quantity of "crude information" into a temporary data store, uses constraint satisfaction problem solving techniques to model the data without full deployment of databases, helps users to browse in this complex data space, and assists them in choosing the best solutions that fit their profile and dynamic criteria [27].

3. Multi-agent framework

A multi-agent intelligent system is utilized in this paper for modeling a comparative shopping mechanism. It is suitable for describing the coordinating and negotiating nature of sellers in a market. Negotiation is a process that takes place between two or more agents who are attempting to achieve goals when they cannot achieve their own original goals. Since these goals may conflict, they have to communicate between themselves to achieve the goals [26]. Multi-agent systems offer a new dimension for coordination and negotiation in an enterprise. Incorporating autonomous agents into the problem-solving process allows improved coordination of different functional unit-defined tasks, both independent of the user and of the functional units under control [2,3,14,16,20,25,33–35,38–40]. Under a multi-agent system, the problem-solving tasks of each functional unit become populated by a number of heterogeneous intelligent agents with diverse goals and capabilities [17,18,22,38,40].

In this paper, we have assumed a system that consists of single buyer (*B-agent*), multiple sellers (*S-agents*), and one negotiator (*Negotiator*). The agents are defined by the following set of characteristics.

$$D_j --- \text{Vector of proposal provided by } S - agent\ j = < e_1, e_2, \ldots, e_n > \quad (1)$$

where e_k, $1 \leq k \leq n$ denotes kth element to negotiate

$$U --- \text{Buyer's utility function} = f(D_j, C, s) \quad (2)$$

where s denotes sensitivity about contextual pressure

$$C --- \text{Vector of contextual information} = < c_1, c_2, \ldots, c_m > \quad (3)$$

where c_l, $1 \leq l \leq m$ denotes lth contextual data

$$UP_j --- \text{Seller } j\text{'s unit profit function} = p_j - UC_j \quad (4)$$

where UC_j denotes Seller j's unit cost function

$$PM_j --- \text{Performance measures of } S - agent\ j. \quad (5)$$

4. System architecture

The overview of our system architecture is shown in Fig. 1. The *Negotiator* is always listening to any *B-agent*, which wants to find comparative goods which are proposed through the selling agents (*S-agent*). The *B-agent* can be downloaded and resides in the buyer's mobile device or possibly on the server. When the buyer goes shopping and finds a candidate product for purchase, he/she may ask his/her own *B-agent* if any other goods with competitive condition exist in other shops. The *B-agent* then submits a new request to the *Negotiator* so that it may introduce some other agents who are interested in proposing the same or similar goods with better condition. The *Negotiator* first selects a set of *S-agents* by querying a self-contained information repository. Secondly, the request from the *B-agent* is streamed to the selected *S-agents*.

The ultimate goal of the negotiation in this system is to realize win–win situations between the *B-agent* and the *S-agent*. The *B-agent* may get more competitive goods than what the buyer is actually seeing at that time. The *S-agent* can increase total sales and profits by encompassing new buyers by a *Negotiator*. The *Negotiator* will also maximize the rate of successful contracts between the *B-agent* and the *S-agent*. To do so, the *Negotiator* should:

- provide information on what a buyer wants as sufficiently and correctly as possible
- encourage the *S-agents* to offer more competitive bids
- bring the last suggestion to the *B-agent* as soon as possible:

The first goal is closely related to how correctly the *Negotiator* or *S-agent* fits buyers' preferences. However, it is natural to assume that both the *Negotiator* and *S-agents* do not know the correct buyers' utility function because it is nearly impossible to have a considerable number of the buyers' profiles ahead of time. However, it would be reasonable that the seller agents may remember the previous bidding results. Therefore, we put a case base to the *Negotiator*. Table 1 shows a representative subset of the property features used in our architecture. Among these features, contextual information such as location, weather, and calendar is acquired from a context database, the context of which is arriving externally.

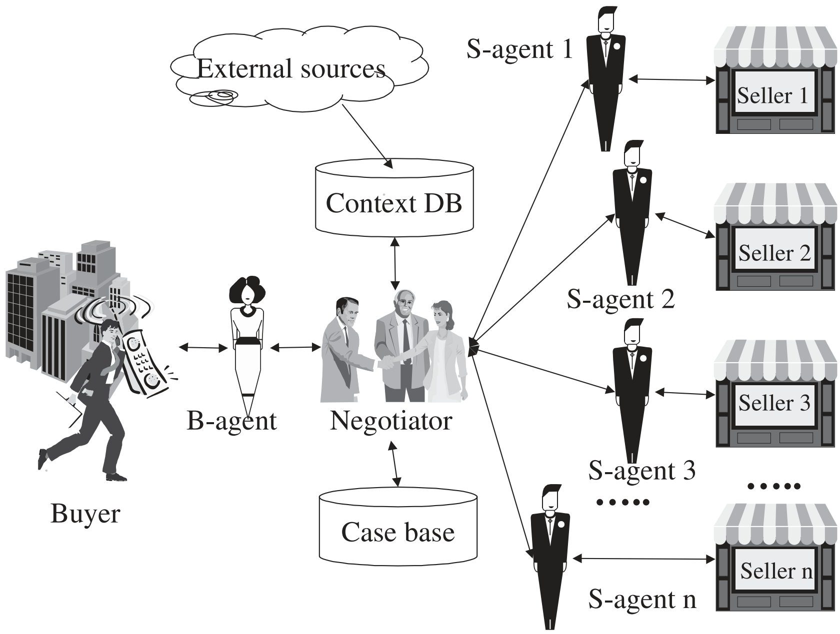

Fig. 1. Overview of the general system architecture.

The similarity between a new problem and a case in memory is computed as follows:

$$\text{Similarity}(t, c) = \sum_{i=1...n} w_i * \text{sim}(t_i, c_i), \quad (6)$$

where t indicates target query, c stands for case, and w denotes weight.

To arrive at the second goal, each of the *S-agents* which receives the request is encouraged to start a cost/benefit analysis to optimize its own unit profit by changing some decision parameters involved in its own cost function:

$$\text{Max TotalProfit} = \text{Sales Volume} \\ *(\text{Unit Price} - \text{Unit Cost})$$

s.t.

Suggesting condition must be delegated by the user. Suggesting condition must be better than any other conditions made by any other *S-agents* and initial condition.

For autonomous negotiation, each *S-agent* is delegated to some extent by its own user. For example, least price allowed by the user is informed and the corresponding *S-agent* can negotiate by modifying its own price condition unless it violates the allowable price level.

Moreover, since the buyer may move around in a mobile setting, the *Negotiator* always checks where the buyer is.

For the last goal, a brokering architecture is basically considered here since it has been known that the brokering theoretically outperforms matchmaking in response to time perspective [10].

Table 1
Subset of the property features in case base

Attribute		Value type
Case_No		Integer
Product_No		Integer
Product_Description		Text
Price		Integer, decimal
Level of quality		Integer, range (1...7)
Preference		Integer
Customer description		Text
Contextual information	Location	Integer
	Weather	Integer
	Calendar	Integer

4.1. Negotiator

The *B-agent* gets a message request from its buyer through a user interface in a mobile device. In the message, data such as current location of the buyer, product name, price, quality are included. Then the *B-agent* asks the *Negotiator* if it can produce a better deal. The *Negotiator* estimates the

```
public class Negotiator
{
   Get buyer's requests;
   Get number of sellers;
   Get sellers' data;
   Get case data;
   Get contextual data;
   Generate a Negotiator;
   Initialize variables;
   runMethod( );

  public static void runMethod()
  {
Initialize quit_sign vector;
    while (!no_candidate_remained)
    {
              do    // updating candidate data or die
              {
                if (candidate[j][quit_sign] == give_me_one_more_chance)
                {
                 candidate[j] = S-agent.run(candidate[j], curr, history);
                  if (candidate[j][quit_sign] == give_me_one_more_chance)
                  {
                   curr = B-agent.run(candidate[j], curr, cases, loc_X, loc_Y, N);
                   candidate[current_winner][quit_sign] = current_winner;
                   }
                 }
                 j++;
              } while (j <= number_of_sellers);

              Check if no_candidate_remained;
    }
     int preference = get_similar_preference(candidate, cases);
     // discount preference according to the distances
     preference -= (Square(X-current_location_x) + Square(Y-current_location_y)/N;
  }

   public static int get_similar_preference(int cand[], int case[][])
   {
      Do
      {
           similarity_price = Square(case[i][price]-cand[price]);
           similarity_quality = Square(case[i][quality] -cand[quality]);
           similarity_total = similarity_price + similarity_quality;
           if (similarity_total < min_similarity_total) change most_similar_total;
      } While (end_of_cases)

         return most_similar_total;
   }
  }
```

Fig. 2. Class description of the negotiator.

buyer's current preference for the product by case-based reasoning, and then retrieves the data set of those buyers who also treat the same or similar products from the buyer table in the information repository. Then *Negotiator* initializes a set of candidates who will participate with the deal. The communication between *B-agent* and *S-agents* is continued until only one candidate remains. The class description of *Negotiator* is shown in Fig. 2.

4.2. S-agent

According to the value of method derived from *Negotiator*, those relevant methods in *S-agent* and *B-agent* begin to run. The class description of the *S-agent* is shown in Fig. 3. An *S-agent* first gets delegation data from the individual database. By fixed interval or special request from its owner, the delegation data may be updated for the time being. The main role of *S-agent* is to provide better conditions for a new subscription from the *Negotiator*. The *S-agent* can autonomously change price or quality level while satisfying given delegation constraints. If a better condition is found, then a new suggestion is prepared and then sent to the *Negotiator*. If not, a quitting sign is issued for withdrawal.

```
Public class B-agent {
  Get initial condition from the user;
  Subscribe to Negotiator;
  Public static int[] run(int candidate[], preference, int X, int Y, int N)
  {
     value = current_optimal_preference;

      if (value < preference)  {
            Change current_optimal_solution;
            quit_sign = current_winner;  }
      else current_optimal_solution remained;
      return current_optimal_solution;
  }
}
```

Fig. 4. Class description of B-agent.

4.3. B-agent

The main contribution of the *B-agent* is to keep the current best condition, compare it with a new

```
Public class S-agent
{
  Get delegation data;
  public static int[ ] run(current_suggestion[], marginal_cost[][])
  {
          int current_margin = current_price – current_unit_cost;
          if (quit_sign == continue)
          {
        for (int p = current_price; p >= price_inferior; p--)
            for (int q = current_quality; q <= quality_superior; q++)
            {
                  unit_cost = current_unit_cost+ (q-current_quality)*marginal_cost[seller_num][quality];
                    new_margin = p - unit_cost;
                    if (new_margin < current_margin && new_margin > max_profit && new_margin >=
                     current_propose)  // if profit is greater than least margin  Store new solution;
             }
            if (no_bettrer_solution)  quit_sign = withdraw_forever
            else
            {
            Change current suggestion;
            Quit_sign = give_me_one_more_chance;
            }
          }
          return new suggestion;
  }

}
```

Fig. 3. Class description of S-agent.

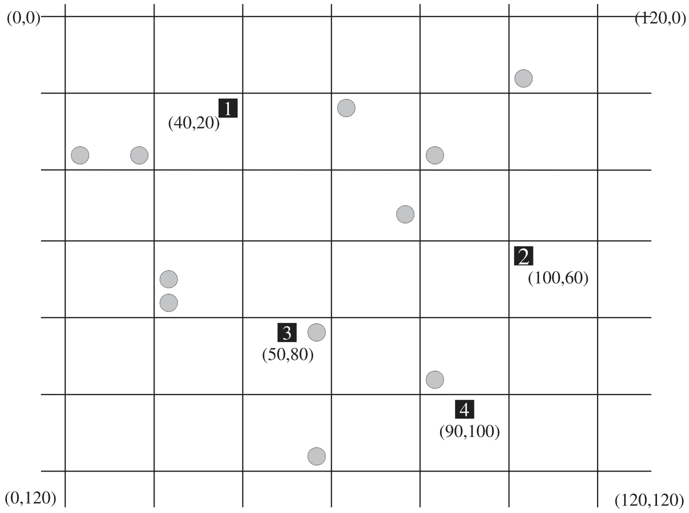

Fig. 5. Illustrative example.

condition arrived from *Negotiator*, and finally, to select the best condition on behalf of the user from among a sequence set of conditions. Since the object-orientation has encapsulation capability, the agents do not need to know each other's internal algorithms. They only pass input and output messages to each other. Similarly, the *B-agent* stands for its client and conducts a pursuit of the client's preference. Its class definition is depicted in Fig. 4.

5. Experiments

5.1. Experimental design

To show the feasibility of the idea of our architecture, let us give an example. A customer is attending a conference and now needs a dinner. He/she has a wireless terminal and will look around at some restaurants within an area ranged from (0,0)–(120,120). Let us assume that there are a total of 15 restaurants, 4 of which have their own selling agents (*S-agents*) that can be accessed by the customer's wireless terminal. The restaurants are tagged as 1, 2, 3, and 4, and they are selling at locations (40,20), (100,60), (50,80), and (90,100), respectively. The locations are shown Fig. 5.

The agents are defined by the following set of characteristics.

$$D_j --- \text{Vector of proposal provided by } S - agent\ j = < p_j, q_j > \tag{7}$$

$$p_j --- \text{price level proposed by } S - agent\ j$$

$$q_j --- \text{level of service proposed by } S - agent\ j$$

$$U --- \text{Buyer's utility function} = f(p_j, q_j, w, c, d_{jt}, s)$$

$$w --- \text{weather } (1-5)$$

$$c --- \text{calendar (time pressure) } (1-5)$$

$$d_{jt} --- \text{distance between buyer and seller } j \text{ at time } t$$

$$s --- \text{sensitivity about contextual pressure} \tag{8}$$

The utility function is unknown to the *S-agents* and even to the *Negotiator*. We assumed that there are N buyers but in order to simplify our experiments, we also assumed that they have the same utility functions. The reason for this simplification is that, as the number of buyers increases, the effects from the differences among the utility functions will be reduced because the functions are randomly

```
Context: (Windy or Rainy, 5 hours left)

R#     Location Price Q     UC    Pref  Status
Round 1
0      (11 , 65)     1583  2      0      737    2
1      (40 , 20)     1500  4      1055   1093   1
2      (100 , 50)    1700  7      1285   450    1
3      (50 , 80)     1450  6      1205   1475   0
4      (90 , 100)    1999  7      1300   348    1
Round 2
0      (11 , 65)     1583  2      0      737    2
1      (40 , 20)     1500  7      1070   1573   1
2      (100 , 50)    1697  7      1285   451    1
3      (50 , 80)     1450  7      1210   1635   0
4      (90 , 100)    1996  7      1300   349    1
Round 3
0      (11 , 65)     1583  2      0      737    2
1      (40 , 20)     1431  7      1070   1650   0
2      (100 , 50)    1628  7      1285   474    1
3      (50 , 80)     1449  7      1210   1635   1
4      (90 , 100)    1927  7      1300   372    1
Round 4
0      (11 , 65)     1583  2      0      737    2
1      (40 , 20)     1431  7      1070   1650   1
2      (100 , 50)    1610  7      1285   480    1
3      (50 , 80)     1431  7      1210   1662   0
4      (90 , 100)    1909  7      1300   378    1
..............
Round 14
0      (11 , 65)     1583  2      0      737    2
1      (40 , 20)     1319  7      1070   1775   0
2      (100 , 50)    1405  7      1285   549    2
3      (50 , 80)     1332  7      1210   1752   1
4      (90 , 100)    1703  7      1300   447    1
Round 15
0      (11 , 65)     1583  2      0      737    2
1      (40 , 20)     1319  7      1070   1775   1
2      (100 , 50)    1405  7      1285   160    2
3      (50 , 80)     1319  7      1210   1787   0
4      (90 , 100)    1690  7      1300   451    1
Round 16
0      (11 , 65)     1583  2      0      737    2
1      (40 , 20)     1279  7      1070   1802   0
2      (100 , 50)    1405  7      1285   30     2
3      (50 , 80)     1318  7      1210   1766   1
4      (90 , 100)    1650  7      1300   464    1
Round 17
0      (11 , 65)     1583  2      0      737    2
1      (40 , 20)     1279  7      1070   1802   0
2      (100 , 50)    1405  7      1285   -12    2
3      (50 , 80)     1290  7      1210   1794   2
4      (90 , 100)    1621  7      1300   474    1
Round 18
0      (11 , 65)     1583  2      0      737    2
1      (40 , 20)     1279  7      1070   1802   0
2      (100 , 50)    1405  7      1285   -26    2
3      (50 , 80)     1290  7      1210   1758   2
4      (90 , 100)    1500  7      1300   514    2

Stop. Shop 1 won.
```

Fig. 6. Example of the interactivity of the agents.

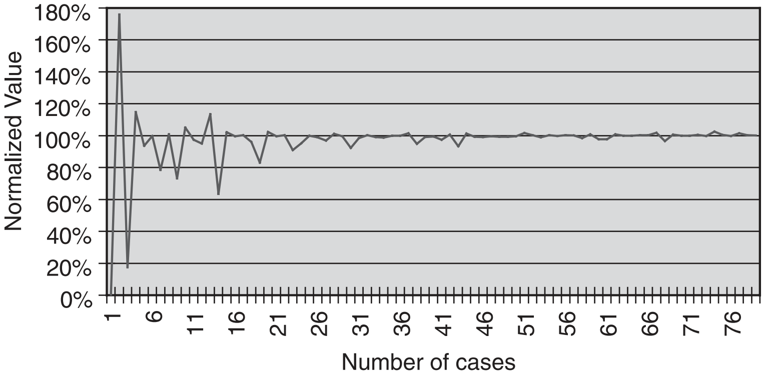

Fig. 7. Convergence of case-based reasoning.

selected and assigned. Therefore, the only thing to additionally consider is to just make *N* personal case bases, which is possible if the *Negotiator* can get the information about users as they register as members. Under such an assumption, in this paper, we have chosen buyer's payoff, sellers' total payoff, sellers'

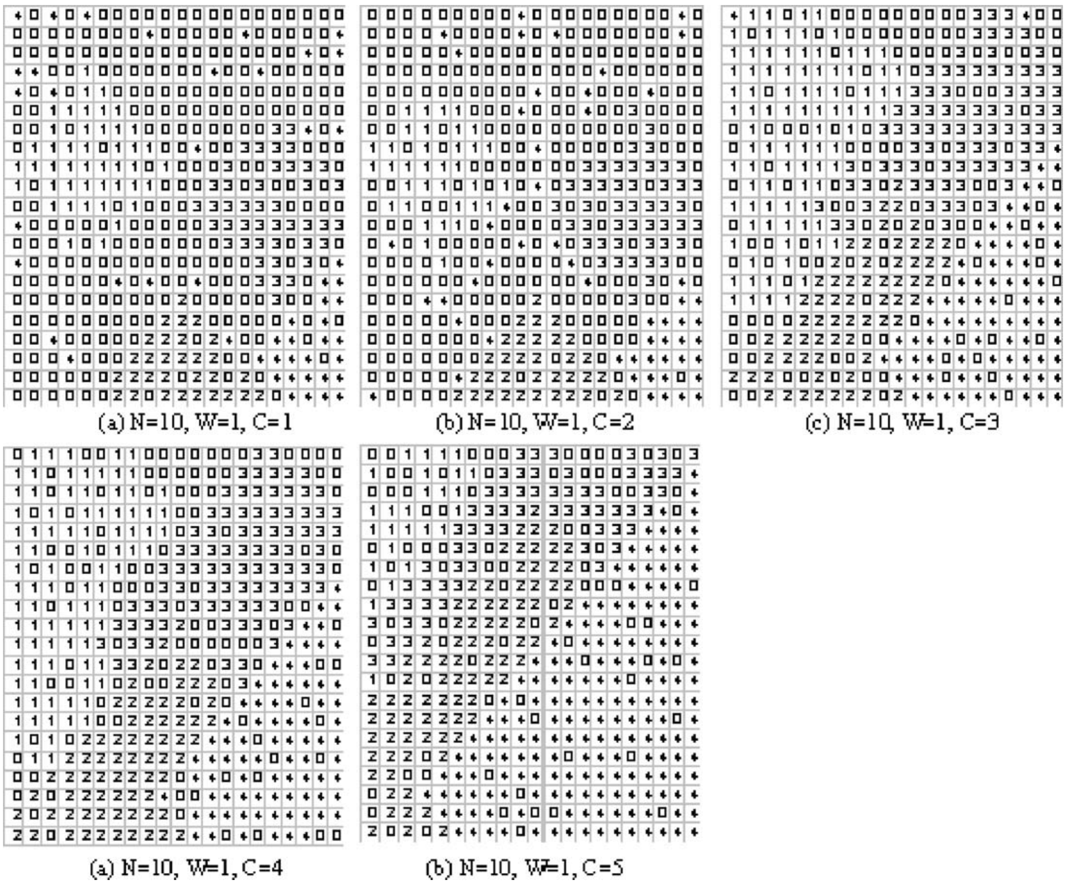

Fig. 8. Distribution of winners.

Table 2
Summary of performances

		Rate of wins	Buyer's payoff	Sellers' payoff
NN	$N=2$	46.4%	196.34	749.85
	$N=6$	46.5%	185.34	713.00
	$N=10$	48.3%	187.87	717.25
PN	$N=2$	57.3%	312.14	830.81
	$N=6$	58.5%	308.45	828.91
	$N=10$	58.9%	297.08	810.97
CN	$N=2$	73.4%	350.75	1106.91
	$N=6$	74.6%	347.17	1094.82
	$N=10$	75.5%	336.79	1073.74

total sales, and rate of win as the performance measures.

5.2. Prototype and analysis

Our proposed prototype was implemented using Java under JDK1.3.1 and experimented on several networked PC platforms. The case base and other data tables are made in Microsoft Access 2000 and linked using ODBC connections. The main goal of our experiment is to investigate the performance of our coordination mechanism. To facilitate our experiment, we assume that the selling agents have been delegated to some extent by corresponding restaurants, and hence may vary their own price level and level of quality to negotiate with the customer. The customer's agent (*B-agent*) sends requests to *Negotiator* to have its customer find the best restaurant.

To analyze the performance of our prototype system, we formulated three types of negotiation: (1) no negotiation (NN), (2) primitive negotiation with only price level (PN) and (3) compound negotiation with quality as well as price level (CN). NN means that the *Negotiator* does nothing but simply deliver requests from the *B-agent* to the *S-agents* and the *S-agents* suggest their own fixed conditions. This kind of negotiation represents traditional content-based shopping mall site: the condition can be updated only manually. Under PN, the agents may vary their own price level autonomously.

Each coordination type was simulated 100 times and the time span of each simulation was 100 periods. At each period, the location and variables of a product provided by an arbitrary off-line shop are produced by random number generation. The following two hypotheses are raised through multi-agents based experiments:

Hypothesis 1: The results by CN will outperform that by NN.

Hypothesis 2: The results by CN will outperform that by PN.

In this paper, we have chosen buyer's payoff (i.e. average preference), sellers' payoff (i.e. total sales), and rate of win as the performance measures. The profit of the *S-agent* is only implicitly considered because we assume that an *S-agent* is a software program purchased by a seller.

The profit of *Negotiator* also is not included in the performance measures because the profit of *Negotiator* is proportional to that of a seller. It is reasonable that *Negotiator* will gain by charging a fee to those sellers as many web sites do, and hence, its sales might be proportional to the frequency of transactions, which will definitely depend on federating sellers' rate of win and/or payoffs. Moreover, win–win situations between sellers and buyers, not just agents, are focused

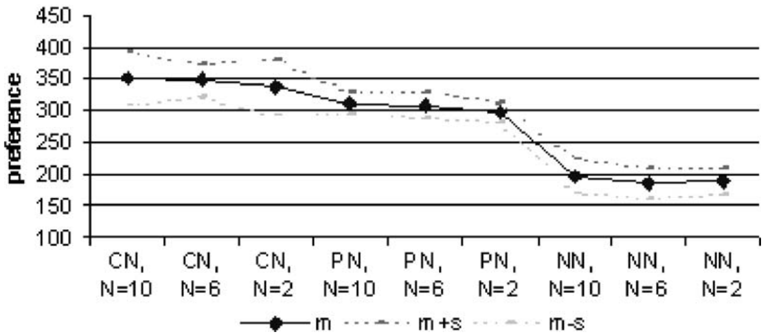

Fig. 9. Buyer's payoff.

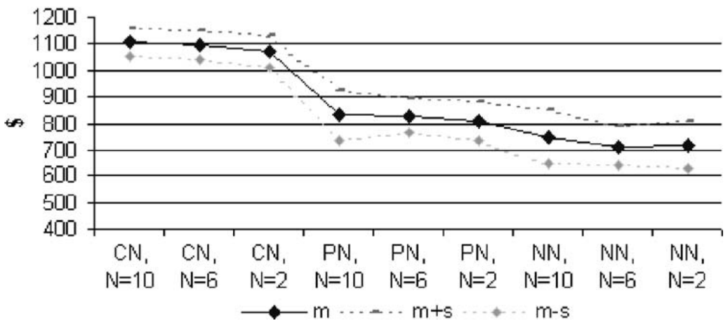

Fig. 10. Seller's payoff.

in performance evaluation. As a result, according to the assumptions, it would be redundant to consider the profits of *Negotiator* or *S-agents* as performance measures. However, it could be worthwhile to consider the agents and negotiator as game players who maximize their own profits and to consider their strategic interaction, which are being actively considered by several researches in the area of game theory.

Fig. 6 shows an example of the interactivity of the agents in negotiation with compound parameters mode. At first, information of a product and location ($R\# = 0$) where a buyer stops is sent to the *S-agent*. In this example, the information vector is < *R#*, location, weather, calendar, price, quality, unit cost, preference, status> = < 0, (11,65), "Windy or Rainy", "5 hours left", 1583, 2, "unknown", 737, 0>. Each of the *S-agents* ($R\# = 1-4$) aims to maximize the profit by changing price and quality. If the value of the decision variables changes, then the cost is also changed by a marginal cost, of which data is contained in the corresponding *S-agent*. However, the decision is subject to a constraint that the cost should not exceed the maximum cost level allowed. If an *S-agent* cannot find an improved suggestion, then the agent quits (status = 2), unless the improved suggestion is delivered to the *Negotiator*, and *Negotiator* then sends it to the *B-agent* so that the agent may compare it with the current suggestion. If the new suggestion is estimated as giving more preference to the buyer, then current winner (status = 0) is changed. The estimation of the preference is based on the knowledge about past cases that are provided by the *Negotiator*.

The Fig. 7 shows how the outcomes of CBR converge into actual value as the number of case increases. The normalized value is calculated as follows (9):

$$\text{Normalized value} = (\textit{Estimated user preference by CBR} / \textit{Actual user preference})*100\% \tag{9}$$

As seen, Fig. 7 shows that the normalized value is rapidly converged to 100%, which means the estimated

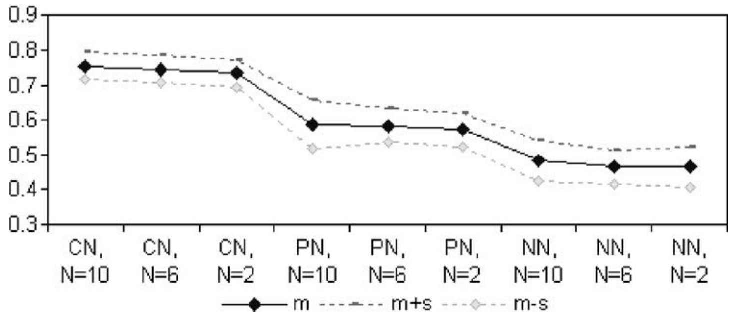

Fig. 11. Seller's rate of wins.

Table 3
Results of statistical test

Hypotheses	Performance measures	Difference	t-value	p-value
Hypothesis 1 ($n=10$)	Buyer's payoff	38.61	2.8593	0.0022***
	Sellers' total payoff	357.06	11.0976	0.0000***
	Sellers' rate of wins	0.17	8.0368	0.0000***
Hypothesis 2 ($n=10$)	Buyer's payoff	238.61	17.6533	0.0000***
	Sellers' total payoff	276.10	9.1224	0.0000***
	Sellers' rate of wins	0.27	13.0526	0.0000***

***$p<0.01$.

value of user preference by CBR is dramatically converged to actual user preference for the time being.

Fig. 8 shows the distribution of winners by changing contextual value: calendar. First, as expected, the number of cells of 0's decreases when the value of C increases: the user has less time pressure of changing restaurant, and hence, he/she can move more easily. Secondly, the area of 1–4 winners is basically located in their own original location. However, the size of the area is quite different from each other, which is caused by bargain power: price and quality. As the time pressure becomes less critical, a seller whose bargain power is relatively strong tends to enlarge its area wider than any other sellers.

Let us investigate the experimental results, which compare the performance of the multi-agents based on the above coordination. To illustrate the performance difference of each coordination type more clearly, we summarize and compare the performance in Table 2 and Figs. 9–11.

Fig. 6 depicts the buyer's payoff of nine alternatives. N is a level of insensibility about distance from a store served by a certain agent and the location where a buyer stands. In other words, when the value of N increases, the sensitivity decreases, which implies the buyer feels indifferent about the distances. The figures clearly show that negotiation with compound parameters (CN) outperforms PN and NN. To show the significance of the performance, Figs. 7 and 8 show that the performance of CN is explicitly better than PN and NN. Table 3 is given to test two hypotheses.

The two null hypotheses provide that the performance models will yield equal results. If p-value is greater than 0.05 or 0.01, then the null hypothesis cannot be rejected statistically. Based on such a principle, we can conclude that the statistical test results for Hypothesis 1 indicate that the null hypothesis is strongly rejected statistically: less than 1% significance levels. We deduce that the active coordination performance outperforms passive coordination.

In the case of Hypothesis 2, in which the negotiation with compound parameter is compared to no negotiation, the null hypothesis is rejected for buyer's payoff, sellers' total payoff, and sellers' rate of wins.

Therefore, we conclude that the method of negotiation with compound parameters yields a better performance than negotiation with single parameter, or no negotiation.

6. Concluding remarks

In this paper, we have described a framework of a context-aware multi-agent intelligent system with multiple parameters and CBR capability for comparative shopping. The framework enables a new competing model: the off- and on-line shops are showing their own products to buyers who are going through off-line shops.

According to the experimental results, as expected, the negotiation with compound items yields a better performance than negotiation with single item and no negotiation, respectively. These show the negotiation feature and the intelligence to autonomously adjust proposals to the buyer's preference by searching for past cases that may represent similar negotiations and anticipating the best condition within the delegation boundary.

In addition to this, based on our simulation, we have found that contextual information such as buyer's weights of distance, weather conditions, and calendar information, may affect the bidding results.

This implies that an ever-changing environment plays an important role in implementing comparative shopping under mobile commerce. Context-awareness does matter and should be considered.

We are now expanding our experiments into more realistic settings from the simulation level. In particular, because the buyers' profiles used in the experiments are artificially created, adopting real profiles may bring out unanticipated experimental results. Additionally, decision-making criteria also must be refined. Even though these restrictions are being left in the proposed system, the proposed mechanism will be expected to open new application areas in context-aware mobile commerce.

References

[1] BargainFinder Agent (Anderson Consulting), http://bf.cstar.ac.com/bf.

[2] A. Bonarini, V. Trianni, Learning fuzzy classifier systems for multi-agent coordination, Information Sciences 136 (1–4) (2001) 215–239.

[3] T. Bui, J. Lee, An agent-based framework for building decision support systems, Decision Support Systems 25 (3) (1999) 225–237.

[4] A. Chavez, P. Maes, Kasbah: an agent marketplace for buying and selling goods, Proceedings of PAAM'96, Practical Application Company, 1996, pp. 75–90.

[5] A. Chavez, A. Moukas, P. Maes, Challenger: a multi-agent system for distributed resource allocation, Proceedings of the First International Conference on Autonomous Agents, Marina Del Rey, ACM Press, 1997, pp. 323–331.

[6] J. Cheng, M.P. Wellman, The WALRAS algorithm: a convergent distributed implementation of general equilibrium outcomes, Computational Economics 12 (1998) 1–24.

[7] S. Clearwater, Market-Based Control: A Paradigm for Distributed Resource Allocation, World Scientific Publishing, Singapore, 1996.

[8] CompareNet, Interactive Buyers Guide, http://www.compare.com.

[9] Dealpilot, The Ultimate Comparison Shopping Engine, http://www.dealpilot.com.

[10] K. Decker, M. Williamson, K. Sycara, Matchmaking and Brokering, Working Paper, CMU, 1996.

[11] R.B. Doorenbos, O. Etzioni, D.S. Weld, A scalable comparison-shopping agent for the world-wide web. The First International Conference on Autonomous Agents. ACM Press, 1997, pp. 64–65.

[12] Excite Shopping, http://www.jango.com.

[13] A. Eyal, T. Milo, Integrating and customizing heterogeneous e-commerce applications, VLDB Journal 10 (1) (2001) 16–38.

[14] J. Hu, M.P. Weliman, Learning about other agents in a dynamic multiagent system, Cognitive Systems Research 2 (1) (2001) 67–79.

[15] J. Kolodner, Case-Based Reasoning, Morgan Kaufmann, San Francisco, 1993.

[16] S. Kraus, J. Wilkenfeld, G. Zlotkin, Multiagent negotiation under time constraints, Artificial Intelligent Journal 75 (2) (1995) 297–345.

[17] C. Lottaz, I.F.C. Smith, Y. Robert-Nicoud, B.V. Faltings, Constraint-based support for negotiation in collaborative design, Artificial Intelligence in Engineering 14 (3) (2000) 261–280.

[18] X. Luo, C. Zhang, H.F. Leung, Information sharing between heterogeneous uncertain reasoning models in a multi-agent environment: a case study, International Journal of Approximate Reasoning 27 (1) (2001) 27–59.

[19] T.W. Malone, R.E. Fikes, K.R. Grant, M.T. Howard, Enterprise: a market-like task scheduler for distributed computing environments, in: B.A. Huberman (Ed.), The Ecology of Computation, Elsevier, New York/Amsterdam, United States, 1988, pp. 177–205.

[20] F.P. Maturana, D.H. Norrie, Distributed decision-making using the contract net within a mediator architecture, Decision Support Systems 20 (1997) 53–64.

[21] S.A. McIlraith, T.C. Son, H. Zeng, Semantic web services, IEEE Intelligent Systems 16 (2) (2001) 46–53.

[22] P.R. McMullen, An ant colony optimization approach to addressing a JIT sequencing problem with multiple objectives, Artificial Intelligence in Engineering 15 (3) (2001) 309–317.

[23] J. Morris, P. Maglio, When buying on-line, does price really matter? Proceedings of the Conference on Human Factors in Computing Systems, ACM Press, 2001, pp. 99–100.

[24] J. Morris, P. Ree, P. Maes, Sardine: dynamic seller strategies in an auction marketplace, Proceedings of the Conference on Electronic Commerce, ACM Press, 2000, pp. 128–134.

[25] H. Nishiyama, W. Yamazaki, F. Mizoguchi, Negotiation protocol for proof of realization of cooperative task in multi-agent robot systems, Proceedings of IEEE International Conference on Systems, Man, and Cybernetics, IEE, 2000, pp. 1685–1690.

[26] S. Parsons, C. Sierra, N. Jennings, Agents that reason and negotiate by arguing, Journal of Logic and Computation 8 (3) (1998) 261–292.

[27] P. Pu, B. Faltings, Enriching buyers' experiences: the SmartClient approach, Proceedings of ACM CHI'2000, ACM Press, 2000, pp. 212–213.

[28] A. Rajaraman, P. Norvig, Virtual database technology: transforming the Internet into a database, IEEE Internet Computing 2 (1998) 55–58.

[29] J. Rosenschein, G. Zlotkin, Rules of Encounter: Designing Conventions for Automated Negotiation among Computers, MIT Press, Cambridge, MA, United States, 1994.

[30] S. Samaras, S. Evripidou, E. Pitoura, A mobile-agent based infrastructure for eWork and eBusiness applications, Proceedings of eWork and eBusiness Conference, 2000, pp. 1092–1098.

[31] Savvy Search, http://www.MEGAGO.com/1/.
[32] E. Selberg, O. Etzioni, Multi-service search and comparison using the MetaCrawler, Proceedings of the 4th International World Wide Web Conference, O'Reilly & Associates, INC Publisher, Boston, USA, December 1995, p. 195.
[33] J.A.A. Sillince, M.H. Saeedi, Computer-mediated communication: problems and potentials of argumentation support systems, Decision Support Systems 26 (4) (1999) 287–306.
[34] M.P. Singh, Multiagent systems, Lectures in Artificial Intelligence 799 (1994) 81–113.
[35] T.J. Strader, F.R. Lin, M.J. Shaw, Information infrastructure for electronic virtual organization management, Decision Support Systems 23 (1) (1998) 75–94.
[36] A.D. Tsalgati, J. Veijalainen, Challenges in model electronic commerce, Proceedings of the 1st International Conference on E-Commerce and Web Technologies, Springer, 2000, pp. 477–486.
[37] C. Tsatsoulis, Q. Cheng, H.Y. Wei, Integrating cased-based reasoning and decision theory, IEEE Expert 12 (4) (1997) 46–55.
[38] M. Ulieru, D. Norrie, R. Kremer, W. Shen, A multi-resolution collaborative architecture for web-centric global manufacturing, Information Sciences 127 (1–2) (2000) 3–21.
[39] M. Wooldridge, N.R. Jennings, Intelligent agents: theory and practice, The Knowledge Engineering Review 10 (2) (1995) 115–152.
[40] D.J. Wu, Software agents for knowledge management: coordination in multi-agent supply chains and auctions, Expert Systems with Applications 20 (1) (2001) 51–64.
[41] J. Youll, J. Morris, R.C. Krikorian, P. Maes, Impulse: location-based agent assistance, Proceedings of the 4th International Conference on Autonomous Agents, 2000.
[42] G. Zacharia, A. Moukas, R. Guttman, P. Maes, An agent system for comparative shopping at the point of sale, in: J.Y. Roger, et al (Eds.), Technologies for the Information Society: Developments and Opportunities, IOS Press, Amsterdam, Netherland, 1998.

Oh Byung Kwon is presently an associate professor at Handong University, South Korea, where he initially joined in 1996. In 2002, he joins Institute of Software Research International (ISRI) at Carnegie Mellon University to perform DARPRA project on semantic web and context-aware computing. He received the MS and Ph.D. degree in Management Information System at KAIST (Korea Advanced Institute of Science and Technology) in 1990 and 1995, respectively. His current research interests include agent technology, mobile commerce, context-aware system development, case-based reasoning, and DSS. He has published various papers in leading information system journals such as Decision Support Systems, Expert Systems with Applications, and Simulation.