An Interest-based Offer Evaluation System for Semantic Matchmakers

Samira Sadaoui and Wei Jiang

Computer Science Department,
University of Regina, Regina, SK, Canada S4S 0A2

{sadaouis, jiang20w}@uregina.ca

Abstract. Matchmaking systems failed to provide the best matched results to individuals. Semantic matchmaking can help the buyer find the requested offers but it is not good enough to find the best offer. In this work, we propose a system that evaluates and sorts the request-matched offers according to the buyers' interests and tastes. To evaluate the offers, we modify the MultiNomial Logit model to produce an interest model that analyzes individual's interests and favors. Our system captures the buyer's interests, builds his interest model, and then returns the best offer. The best offer denotes the highest interest value to the buyer. Through a case study, we present in detail the phases of our offer evaluation process.

Keywords: Interest model, multiple offer attributes, best matched offer, Self Organizing Map (SOM), MultiNomial Logit (MNL).

1 Introduction

Matchmaking is the online process through which buyers and sellers trade goods or services. Most of the matchmaking systems are semantic-based. Research on ontology lead the early semantic matchmaking systems to understand and process the purchasing requests much better [1, 2, 3]. Nevertheless, with the blooming of e-commerce and e-services, buyers can obtain more and more request-matched offers. It is really time consuming for buyers to browse, evaluate and sort all the candidate offers in order to find the best offer. Today determining the best offer is more important than before for any matchmaker. Recent matchmakers are trying to determine the best offer by using the semantic ranking [4, 5, 6, 7, 8]. Most semantic ranking algorithms examine the similarity of inputs, outputs, preconditions and effects. Yet all existing matchmakers failed to bring the best matched results to individuals. Indeed, they do no guarantee that the best offer will be purchased by the buyer since his specific interests and tastes are ignored. Without studying human interests and only relaying on the semantic matching and ranking, matchmakers cannot recognize the differences between buyers' favors and needs.

Researchers realized that a better matchmaking system "*could quicken the trend toward personalization*" [9]. Matchmaking based on semantic can help the buyer find

the requested offers but it is not good enough to find the best offer. Consequently, we develop a system that evaluates and sorts the request-matched offers according to the buyers' specific interests and tastes. In our system, the best offer denotes the highest interest value to the buyer. Analyzing individual interests along with the semantic matching brings better results to each individual buyer. As illustrated in Figure 1, first the buyer submits to our system a purchasing request which is then sent to a connected semantic matchmaker. The latter returns a list of request-matched offers. To sort these offers according to the buyer's interests, our system performs the following tasks: cluster the values of each offer attribute, interact with the buyer to take into account his interests and tastes, calculate the attribute's interest weight and interest rate, build the interest model based on interest weights and rates, evaluate and sort the offers according to the buyer's interest model.

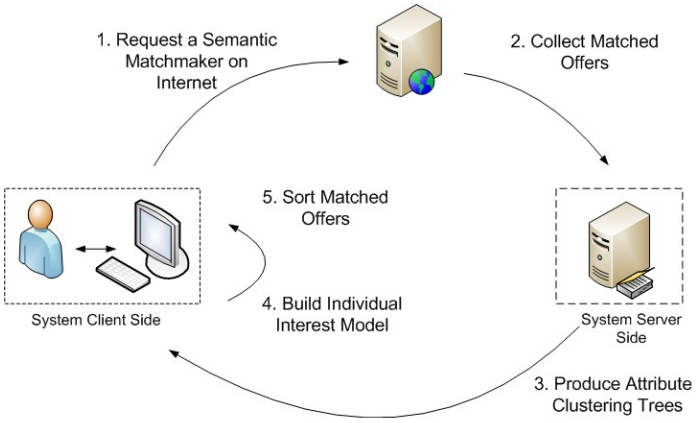

Fig. 1. System Process Overview

In order to take into account individual's interests, we need to utilize the clustering technology called Self Organizing Map (SOM) [10, 11]. As a neural-network approach, SOM is employed to cluster high-dimensional inputs onto lower-dimensional outputs. The reason of using SOM in our work is that the offer attributes may be complex or contain high-dimensional data, such as the attributes of our case study. Furthermore, to define the interest model specifically for each buyer, we modify the MultiNomial Logit (MNL) model [12]. MNL is widely used in commerce to study human shopping behaviors [13]. Nevertheless, MNL suggests a model for a group of people and needs an appropriate sample data. The good thing is that we are able to modify MNL to analyze individual's needs alone and without considering a sample data.

2 Related Work

In the early time of e-commerce, matchmakers focused on mapping the offer attribute values [14]. Requests and offers can be expressed in different schemas and words

even when representing the same semantic meaning. This causes matchmakers to be blind to some potential offers. To solve this problem several semantic matching models, based on ontological technologies, have been proposed [1, 2, 3]. In [1], matching the offers is based on the similarity of the request and services. [2] uses logical relationships to map the offers with the request. [3] focuses on the semantic matching using a platform-independent framework called UDDI. These matchmakers return an unranked list of offers.

To find the best offer, recent matchmakers evaluate service description [4, 5], service constraints [6], service process [7] or both [8]. The best offer denotes the highest semantic matching degree. These matchmakers map the functional properties of offers with the request's functional description. However, these matchmakers cannot explain why sometimes a buyer prefers an offer different from the returned best offer. To address this issue, non-functional matching methods [15, 16] have been introduced by following a matching standard like Qos [17]. These papers argue that the buyer's choice is caused by other criteria often referred to as non-functional properties.

All these matchmakers are based on matching the query words but not on matching individual's interests. Our goal is to build the interest model specifically for the buyer and then find the best matched offer which is the closest to the buyer's real needs.

3 An Example

Our case study consists of purchasing computers based on 2-dimensional attributes: CPU and Price. We have here a company which submits the following inventory query:

> *Request computers with CPU > 1.5 GHz, Price < $3500 for the first ten purchased computers, and Price < $3000 for the next ten.*

To formulate the requests and offers, matchmakers utilize well known web service languages, such as WSDL, SWSL and WSML, which offer a high degree of flexibility and expressiveness. Thus, we translate the purchasing request to for example WSDL following the structure defined in [18] (cf. Figure 2). We assume the connected semantic matchmaker returns the candidate offers given in Table 1 where CPU contains high dimensional values and Price has two ranges.

In the next phases, our system will evaluate all the candidate offers of Table 1 in order to help the company find the best supplier which might become its long-term business partner.

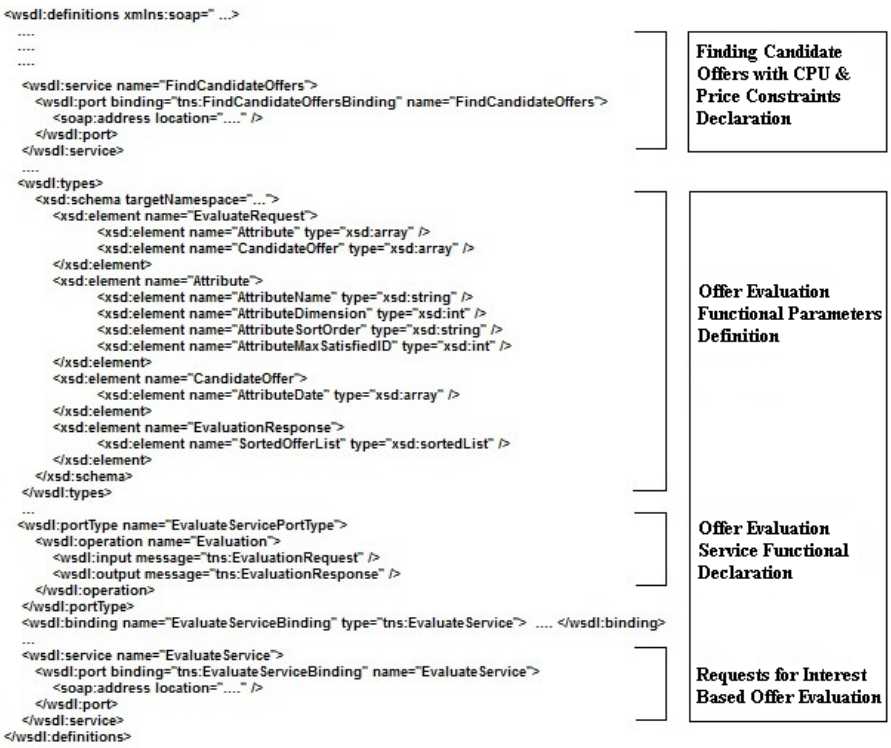

Fig. 2. Formatting the Purchasing Request

Table 1. Candidate Offers

Supplier ID	CPU (GHz)	PriceRange1 ($) For the First 10 Items	PriceRange2 ($) For Item 11 to 20
1	2.5	1100	799
2	2.2	470	370
3	2.5	600	500
4	2.33	1100	800
5	2.4	999	799
6	2.5	1030	830
7	2.66	2500	2200
8	2.3	1000	880
9	2.2	420	400
10	2.4	950	900
11	2.8	1200	1150
12	(1.9, 1.9)	800	700
13	(3.0, 3.0)	2900	2600
14	(1.8, 1.8)	680	680
15	(3.2, 3.2)	3200	2500

4 Attribute Data Clustering

Our system first determines all the attributes from the purchasing request. For each attribute, it extracts all its values from the candidate offers and stores them in a single table. Our system can now cluster the values of each attribute. The purpose of this clustering is to be able to take into account the buyer's interests. So the buyer can select one of the clustering to represent his most interested area. In Figure 3, we define the clustering function called *ClusterAttributeData()* which is based on the algorithm Self Organizing Map (SOM) [11] given in Figure 4.

We apply SOM to recursively divide a large clustering into three sub-clustering until there are less data in the clustering. During the learning time, a set of Learning Vector Quantizations (LVQs) is tuned towards the input attribute data. SOM applies competitive-learning given in the steps 3.1.1 and 3.1.2 of Figure 4. In SOM function: t is the learning time; *CreateLVQ()* is a function that generates three random LVQ m_i in the range of *DataAttribute*; *alpha (t)* controls the learning loop and is the learning rate function which is decreased by learning time t; m_c is the closest LVQ to the selected data x; $h_{ci}()$ is the "neighborhood" function that updates LVQ in the learning time [11].

```
void ClusterAttributeData(DataAttribute: Array)
{1. SOM(DataAttribute, A1, A2, A3);
    //Cluster all data into 3 groups
 2. ArrangeClustering(A1, A2, A3);
    //Arrange clustering in ascending order
 3. for i = 1 to 3
    if  (IsLargeEnough(Ai))
         ClusterAttributeData(Ai);
         //Cluster each sub-group}
```

Fig. 3. Clustering Algorithm of Attribute Data

```
void SOM(DataAttribute: Array, A1: Array, A2: Array,
A3:   Array)
{1. var  t = 1;    //Initialize  learning time
 2. CreateLVQ(DataAttribute, m1,m2,m3);//Create LVQ mi
 3. while (alpha(t) is not too small)//Decrease by time
    {3.1 while (PickUpValue (DataAttribute, x))
                //Select x in the data set
      {  3.1.1  ||x - mc||  =   min{||x - mi||};
                //Find closest LVQ mc
         3.1.2  mi(t+1) = mi(t) + hci(t)[x(t)-mi(t)];
                //Update mi during learning
      }
     3.2 t = t+1;//Update time
    }}
```

Fig. 4. SOM Algorithm

Example. Figures 5 and 6 illustrate respectively the CPU and Price clustering (represented as a tree structure).

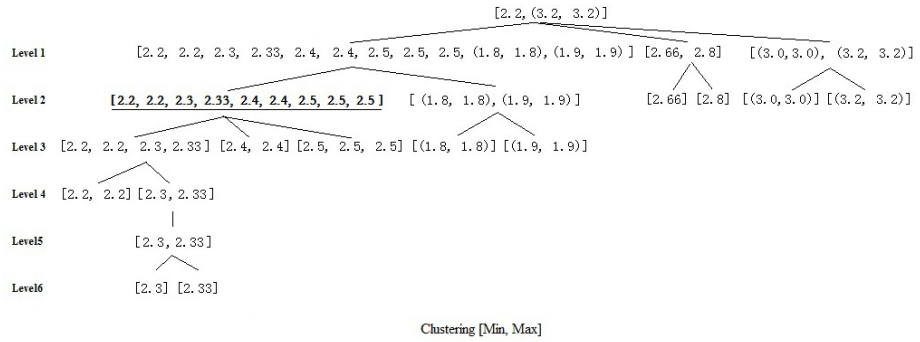

Fig. 5. CPU Data Clustering

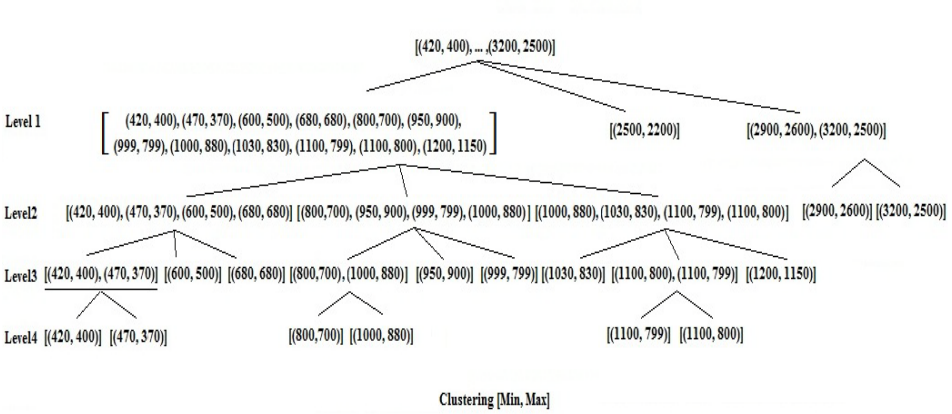

Fig. 6. Price Data Clustering

Our system displays the CPU and Price clustering to the company. The latter can now select the most interested clustering for each attribute. These selections are performed through the GUI of Figure 11. With these selections, our system knows the range and depth of the company's needs. The selection information, representing the company's purchasing interests, will help the system to create the interest model in the following phases.

4 Interest Weight Computation

An interest weight denotes the degree of importance of an attribute in a matching. It is related to the depth and range of the buyer's selection. The smallest and deepest clustering shows the best interest of the buyer. In order to produce the attribute's weight, we use the interest-weight coefficient which contains the buyer's interest in one attribute. We create Formula **(1)** to compute the interest-weight coefficient of an attribute called *k*: *K* is the attribute set, *DataAttribute* is the data range of k, *SelectedClustering* is the buyer's selected clustering for *k*, *SelectedLevelTree* is the selected clustering level, and *TotalLevelTree* is the number of levels of the clustering tree.

$$IW\_coe_k = \frac{DataAttribute_k}{SelectedClustering_k} \cdot \frac{SelectedLevelTree_k}{TotalLevelTree_k} \quad k \in K \tag{1}$$

This coefficient has to be compared with the other attributes' coefficients (cf. Formula **(2)**). An attribute with a larger interest weight coefficient gets a larger interest weight compared to the other attributes. This means we can finally link the interest weights with the buyer's selection.

$$IW_k = \frac{IW\_coe_k}{\sum_{k=1}^{K} IW\_coe_k} \qquad k \in K \tag{2}$$

Example. We suppose the company chooses the CPU clustering [2.2, 2.5]. This selection is at level 2 of the 5-level CPU tree and all CPU data are in the range of [2.2, (3.2, 3.2)]. So, the CPU interest weight coefficient is calculated as 4.4667 with Formula **(1)**. We perform the same calculation for the Price attribute with a coefficient of 44.8125. According to Formula **(2)**, we can get all the attribute interest weights (as shown in Figure 13). For example, CPU interest weight is the following:

$$IW_{CPU} = \frac{4.4667}{4.4667 + 44.8125} \approx 0.0906$$

Based on the interest weights of Figure 11, we can see that Price is much more important than CPU. Such interest feature will help the company to select its best supplier.

5 Interest Rate Function Generation

The goal here is to produce the interest rate of each attribute. To do this, we need to define an interest rate function. A linear function is usually used to measure the attributes' rates [1, 13, 14]. In some cases, linear utility functions cannot assign weights to attributes in order to make an offer as the best one.

In order to solve this problem, we use the un-linear sigmoid function $\varsigma_k(x) = \frac{1}{1+\exp^{-x}}$. We believe the sigmoid function is the closest function to human natural interest change. In our work, we use the sigmoid function to simulate each

attribute interest rate. Here x denotes the value of an attribute and y its interest value. In Figure 7, we show that x of the sigmoid function has less changes in the two intervals [-∞, -2] and [2, +∞]. The Sigmoid function in these two intervals can be considered as a linear function with an acceptable standard error. Meanwhile the interval [-2, 2] is a quickly changeable area. A buyer's selected attribute clustering contains a specific interest for this attribute. In order to represent such interest in our interest rate function, we need to bind the buyer's selected clustering into the quickly changeable interval [-2, 2].

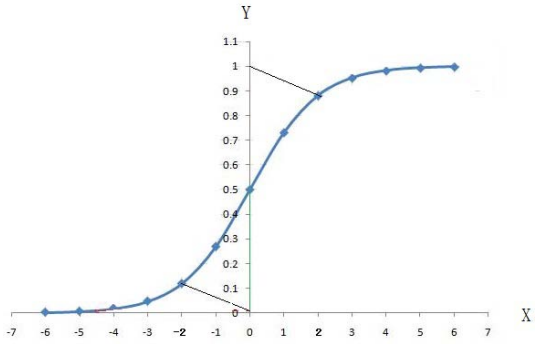

Fig. 7. Sigmoid Function and Value Change

We employ LVQ as the center of our interest rate function since LVQ can be considered as the density center of the attribute data. Attribute data [AL, AR] can be distributed accordingly, and the selected clustering [L, R] is bound into the interval [-2, 2]. LVQ point can be either in the selected clustering [L, R] or outside. If LVQ is inside the clustering, we divide the interest rate function into two functions (cf. Formula **(3)**) where α_L is generated when binding [L, LVQ] into the interval [-2, 0], α_R is generated when binding [LVQ, R] into [2, 0]**,** *Sign* is +1 or -1 w.r.t ascending or descending order of the clustering.

$$\varsigma_{Attribute}(x) = \begin{cases} \dfrac{1}{1+\exp^{[\alpha_L \cdot Sign \cdot (x - \mathrm{LVQ})]}} & x \in [AL, \mathrm{LVQ}] \\ \dfrac{1}{1+\exp^{[\alpha_R \cdot Sign \cdot (x - \mathrm{LVQ})]}} & x \in [\mathrm{LVQ}, AR] \end{cases} \tag{3}$$

$$\alpha L \cdot (x - \mathrm{LVQ}) = -1 \cdot (-2 - 0) \quad \text{when } x = L$$
$$\alpha L = 2 / (L - \mathrm{LVQ})$$
$$\alpha R \,.(x - \mathrm{LVQ}) = -1 \cdot (2 - 0) \quad \text{when } x = R$$
$$\alpha R = -2 / (R - \mathrm{LVQ})$$

Example. We can now generate the interest rate functions for CPU and Price by binding all the selected clustering into the quickly changeable area. According to the CPU attribute tree, the selection [2.2, 2.5] has the LVQ of (2.4767, 0.3189). To be

able to process high dimensional values, we use the distance between attribute data and the best attribute data. Here, we believe CPU clustering [3.2, 3.2] is the best data.

Table 2. Distances of High-Dimensional Data

x	BestAttributeData	Distance(x, BestAttributeData)
2.2 ***	(3.2, 3.2)	3.3526
2.5 **	(3.2, 3.2)	3.2757
(2.4767,0.3189) *	(3.2, 3.2)	2.9705

***Min selection value (L), **Max selection value (R), *LVQ

According to the distance calculation in Table 2, LVQ is closer to the best data than any other data in the selected clustering. Consequently, the following interest rate function for attribute CPU has only α_L. Figure 8 displays the company's interest rate for CPU.

$$\varsigma_{CPU}(x) = \frac{1}{1+\exp^{[\alpha_L \cdot Sign\cdot(Distance(x,BestAttributeData)-Distance(\mathrm{LVQ},\mathrm{BestAttributeData}))]}}$$

$$= \frac{1}{1+\exp^{5.2342\cdot[Distance(x,(3.2,3.2))-2.9705]}} \quad x \in [2.2,(3.2,3.2)]$$

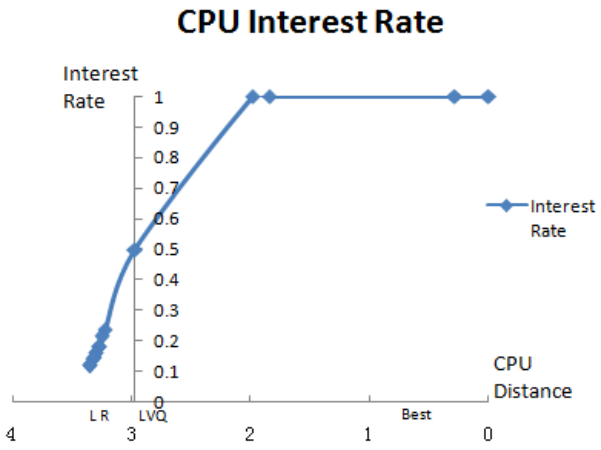

Fig. 8. Interest Rate Function for CPU

We suppose that the company selected the Price clustering [(420, 400), (470, 370)] which is then bound into the area [-2, 2]. After data binding, we produce the following interest rate function where LVQ of 39.88 is its center point. Based on this function, we can easily get the company's interest rate for Price attribute as shown in Figure 9.

$$\varsigma_{\Pr ice}(x)=\begin{cases}\dfrac{1}{1+\exp^{[\alpha_L\cdot Sign\cdot(Dis\tan ce(x,BestAttributeData)-Dis\tan ce(\mathrm{LVQ,BestAttributeData}))]}}\\ \dfrac{1}{1+\exp^{[\alpha_R\cdot Sign\cdot(Dis\tan ce(x,BestAttributeData)-Dis\tan ce(\mathrm{LVQ,BestAttributeData}))]}}\end{cases}$$

$$=\begin{cases}\dfrac{1}{1+\exp^{0.0502\cdot[Dis\tan ce(x,(420,400))-39.88]}}\\ \dfrac{1}{1+\exp^{0.1085\cdot[Dis\tan ce(x,(420,400))-39.88]}}\end{cases}\quad x\in[(420,400),(3200,2500)]$$

Fig. 9. Interest Rate Function for Price

6 Offer Evaluation with the Interest Model

MNL model expresses the utility for a group of people choosing an item. The utility function for an individual in a population includes the deterministic and random components as follows [13]:

$$U_{nj}=\sum_{k=1}^{K}b_k\cdot x_{njk}+\varepsilon_{nj},\quad j\in C \tag{4}$$

where

- U_{nj} is the utility for buyer n selecting item j.
- $(b_k \cdot x_{njk})$ *is* the "representative" taste of the population. This component consists of K observed deterministic features x_{njk}; b_k is the weight for each feature x_{njk}.
- ε_{nj} is the individual taste for the item j.
- C is the set of items.

We now adapt the MNL formula to define the interest model for each individual. First we consider the deterministic features as the offer attributes. Furthermore, ε_{nj} can be decomposed into K attributes since ε_{nj} is the total alternative with K features. This means the evaluation of the attributes may be different according to the interests of each individual (cf. Formula (5)). At the market level, the individual taste ε is brought into the population model as a random utility. Since it comes from individuals and its value is random, ε is usually removed when users generate the MNL model. However, in our system individual taste becomes important.

$$U_{nj} = \sum_{k=1}^{K} b_k \cdot (x_{njk} + \varepsilon_{njk}) \quad j \in C \tag{5}$$

In order to produce the interest model, our system calculates the interest weights, IW, and simulates the interest rates, IR. IW denotes the interest weight b_k and IR the interest rate value $x_{njk}+\varepsilon_{nj}$. We propose Formula (6) to build the interest model IM for each individual. IM is the degree of interest of the buyer purchasing an offer j with K attributes. Based on the consumer theory, an individual seeks to maximize his utility in each purchasing behavior.

$$IM_{(j)} = \sum_{k=1}^{K} (IW_{jk} \cdot IR_{jk}) \quad j \in C \tag{6}$$

Example. After our system gets the interest weights and interest rate functions for the two attributes, it generates the interest model (IM) for the company as follows.

$$IM_{Company}(Supplier) = 0.0906 \cdot \varsigma_{CPU}(CPU) + 0.9094 \bullet \varsigma_{Price}(Price)$$

For example, we show below how the interest model calculates the interests for the first two suppliers:

$$\begin{aligned} IM_{Company}(Supplier1) &= 0.0906 \cdot \varsigma_{CPU}(2.5) + 0.9094 \cdot \varsigma_{Price}[(1100,799)] \\ &= 0.0906 \cdot 0.1684 + 0.9094 \cdot 6.2583E\text{-}36 \\ &= \mathbf{0.0189} \\ IM_{Company}(Supplier2) &= 0.0906 \cdot \varsigma_{CPU}(2.2) + 0.9094 \cdot \varsigma_{Price}[(470,370)] \\ &= 0.0906 \cdot 0.1192 + 0.9094 \cdot 0.1192 \\ &= \mathbf{0.1192} \end{aligned}$$

So, we got an interest rate of 0.0189 for Supplier1's offer and 0.1192 for Supplier2's offer. Based on these values, we can conclude that Supplier2 has a higher chance than Supplier1 to be the company's partner. With our interest model, we can evaluate all the candidate offers of Table 1. Table 3 shows that Supplier9 is the best supplier for the company.

Table 3. Sorted Offers for the Company

Supplier ID	CPU (GHz)	PriceRange1	PriceRange2	Interest
9*	2.2	420	400	0.8118
2	2.2	470	370	0.1192
12	(1.9, 1.9) *	800	700	0.0152
13	(3.0, 3.0) *	2900	2600	0.0127
14	(1.8, 1.8)*	680	680	0.0137
15	(3.2, 3.2)*	3200	2500	0.0152
11	2.8	1200	1150	0.0173
7	2.66	2500	2200	0.0122
3	2.5	600	500	0.0152
6	2.5	1030	830	0.0137
1	2.5	1100	799	0.0189
5	2.4	999	799	0.0903
10	2.4	950	900	0.0906
4	2.33	1100	800	0.0900
8	2.3	1000	880	0.0906

**: best offer with the max interest degree*

7 Design and Implementation

We developed our system with a distributed architecture as illustrated in Figure 10. The buyer interacts with the client side via the *GUI*. After the buyer submits his request, the *GUI* passes it to the connected semantic matchmaker. On the server side, the *OfferManager* component: (1) collects the request-matched offers and stores them in the *ContentOffer* database, (2) analyzes the request and offers to extract the attributes and their values, and (3) stores them in the *OfferAttribute* database. For each attribute, the *AttributeDataClustering* component clusters its values and sends its clustering tree to the client side. *GUI* helps the buyer to select the most interested clustering (cf. Figure 11). Once the buyer's selection is completed, the *InterestModelCreator* component is called to build the buyer's interest model. It first passes all the selected clustering and the whole attribute clustering trees to *InterestWeightCalculator* and *InterestRateFunctionCreator*. For each attribute, *InterestWeightCalculator* returns the interest weight coefficient and interest weight (cf. Figure 12), and *InterestRateFunctionCreator* the interest rate function. The *OfferEvaluator* component applies the generated interest model on the candidate offers, and returns to the buyer the list of offers sorted by interests. In Figure 13, we show for instance the offer evaluation process on the client side.

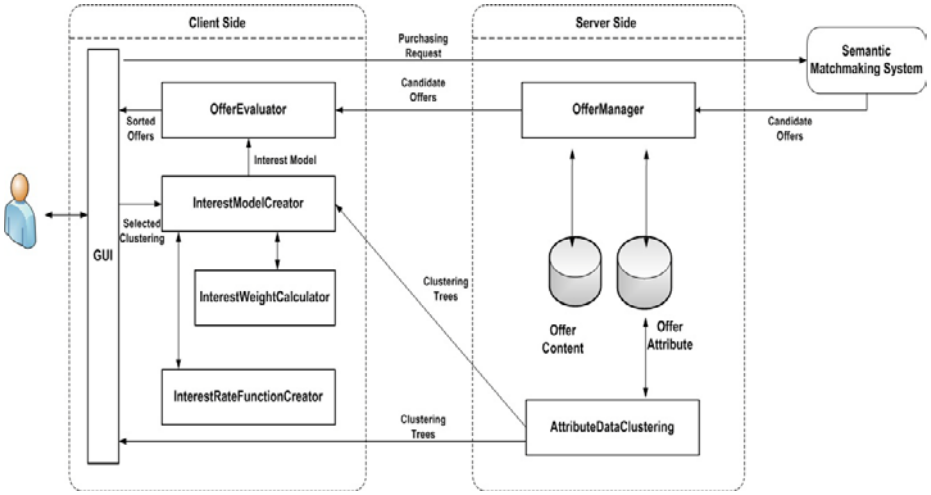

Fig. 10. System Top-Level Architecture

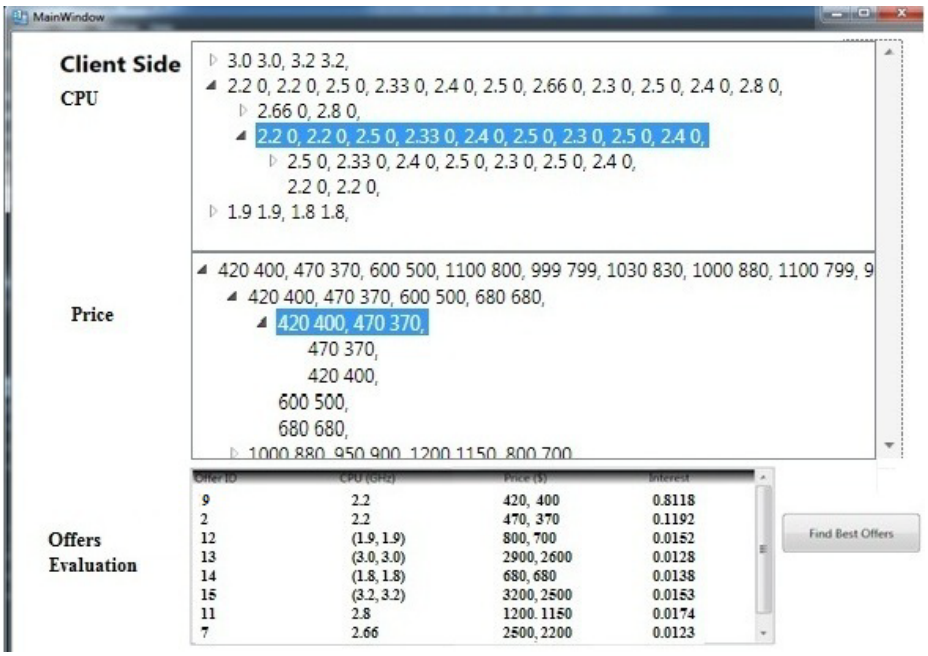

Fig. 11. Selecting CPU and Price clustering

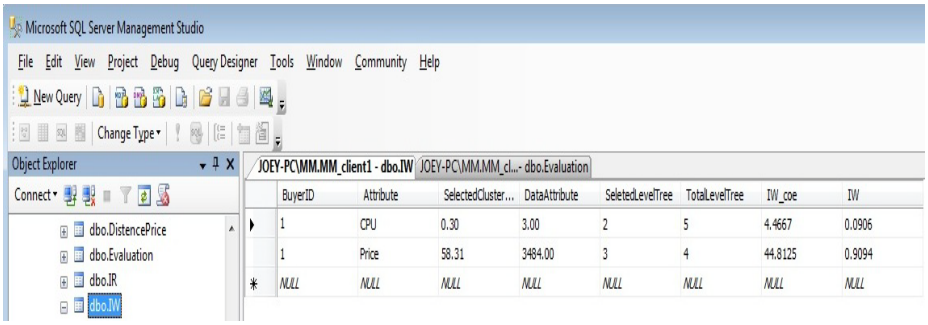

Fig. 12. Calculating the Interest Weights for CPU and Price

We implemented our system using Visual Studio C# (on the .net 3.5 framework) and the SQL Server 2008. Figures 14 and 15 show the class diagrams of the client and server side programs. We created two separate databases sources, called *serverDB* and *clientDB*, to support server and client side programs. These two classes contain all the necessary functions about database processing and data binding. The other classes are created with a window interface by using software MS Blend3, the interface developing tool for Windows form application. The classes *ServerWindow* and *ClientWindow* contain multi-thread and network communication functions.

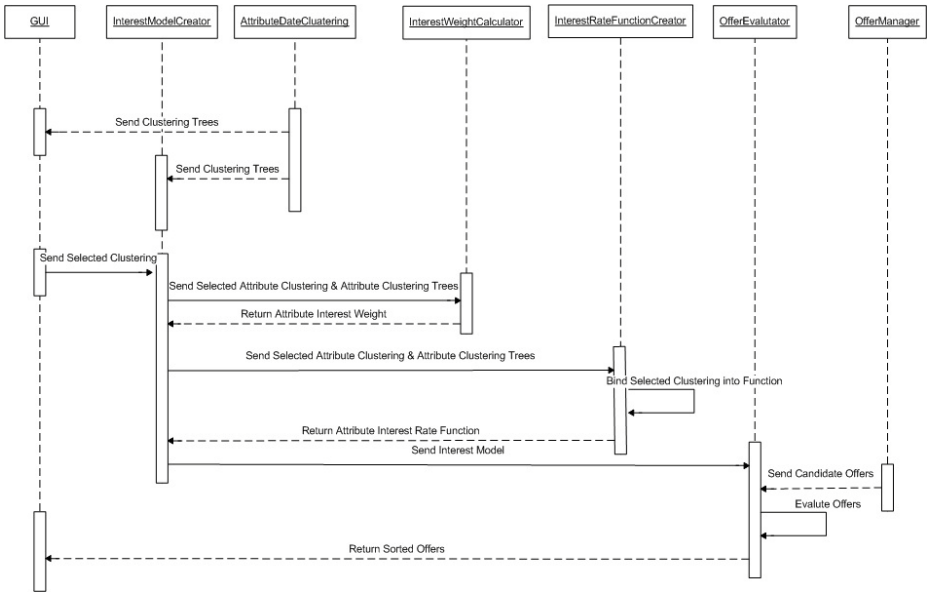

Fig. 13. Client Side: Offer Evaluation

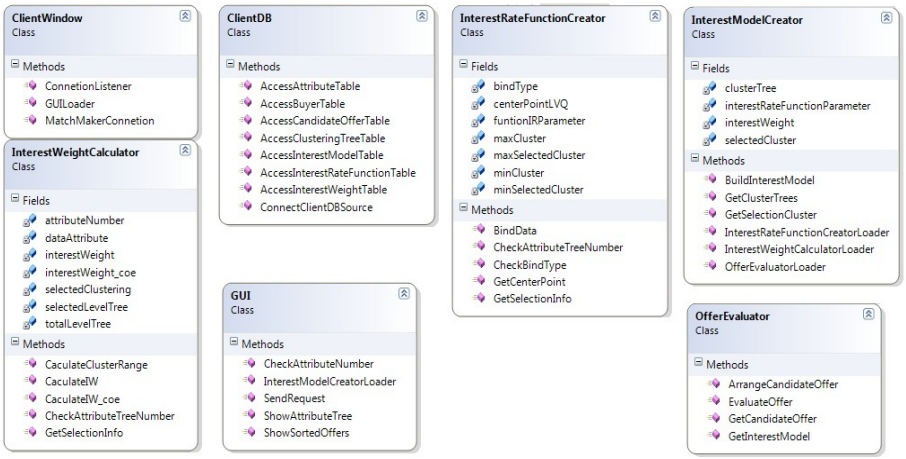

Fig. 14. Server Side Class Diagram

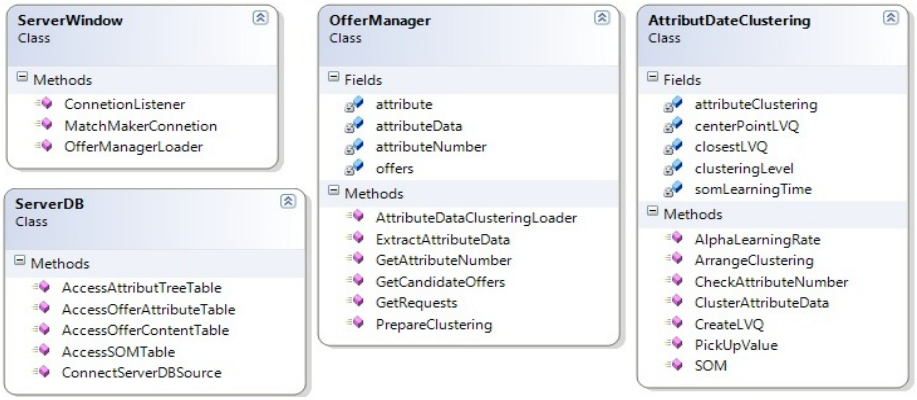

Fig. 15. Client Side Class Diagram

8 CONCLUSION AND FUTURE WORK

In this paper, we showed the benefits of sorting the request-matched offers according to the buyer's interests and needs. Our interest model provides a solution to existing matchmaking systems and avoids the linear matching problems. Adopting an economic method, we produced a simple and automated model to determine the best matched offer based on the buyer's selections.

One possible direction of this work is to include the interest learning [19] in our system. The main purpose of this learning is to update the interest model to fit the buyer's interests instantly. A learned interest model will be able to determine the best offer in these two situations: the buyer shifts his interests, or new offers are added in our database.

References

1. Hahn, C., Nesbigall, S., Warwas, S., Zinnikus, I., Klusch, M., and Fischer, K. Model-Driven Approach to the Integration of Multi-Agent Systems and Semantic Web Services. Enterprise Distributed Object Computing Conference Workshops. IEEE. 314-324 (Sept. 2008)
2. Wang, H., and Li. Z. Z.: A Semantic Matchmaking Method of Web Services Based on SHOIN^+ (D)*. Services Computing. IEEE. 26-33 (Dec. 2006)
3. Kawamura, T., Hasegawa, T., Ohsuga, A., Paolucci, M., and Sycara, K.: Web services lookup: a matchmaker experiment. IT Professional. IEEE. Vol. 7, No. 2, 36-41 (Mar-Apr 2005)
4. Qiu, T., and Li, P. F.: Web Service Discovery Based on Semantic Matchmaking with UDDI. In Proceedings of the The 9th International Conference. Young Computer Scientists. 1229-1234 (Nov. 2008)
5. Bai, D. W., Liu, C. C., Peng, Y. and Chen, J. L.: Web Services Matchmaking with Incremental Semantic Precision. Wireless Communications, Networking and Mobile Computing. IEEE. 1-4 (Sept. 2006)
6. Bai, D., Fei, A. G., and Cai, S. F.: Semantic Matchmaking of Web Services Constraint Conditions. Wireless Communications, Networking and Mobile Computing. IEEE. 1-5 (Sept.2009)
7. Huang, R., Zhuang, Y. W., Zhou, J. L., and Cao, Q. Y.: Semantic Web-based Context-aware Service Selection in Task-computing. In Proceedings of the WMSO '08 International Workshop. 97-101 (Dec. 2008)
8. Bellur, U., and Vadodaria, H.: Web Service Ranking Using Semantic Profile Information. In Proceedings of the ICWS 2009. IEEE. 872-879 (July 2009)
9. Essex, D.: Matchmaker, matchmaker. ACM Communications, ACM. Vol. 52, 16-17 (May 2009),
10. Kohonen, T.: The Self-Organizing Map, 3rd Edition. Springer (2001)
11. Chen, Y. Y., and Young. K. Y.: Applying SOM as a Search Mechanism for Dynamic System. Decision and Control, IEEE. 4111- 4116 (Dec. 2005)
12. McFadden, D.: Economic Choices. American Economic Association. American Economic Review. Vol. 91, No. 3, 351-378 (Jun. 2001)
13. McFadden, D., and Zarembka, P.: Conditional logit analysis of qualitative choice behavior. Academic Press. Frontiers in Econometrics. 105-142 (1974)
14. Ha, S. H., and Park, S. C.: Matching buyers and suppliers: an intelligent dynamic exchange model. IEEE. Intelligent Systems. Vol. 16, No. 4, 28- 40 (Jul-Aug 2001)
15. Wang, X., Vitvar, T., Kerrigan, M., and Toma, I.: A QoS-aware selection model for semantic web services. In Proceedings of the 4th Int. Conference on Service-Oriented Computing. 390–401 (2006)
16. Yu, Q., and Reiff-Marganiec, S.: Non-functional property based service selection: a survey and classification of approaches. In Proceedings of the Non Functional Properties and Service Level Agreements in SOC Workshop (Nov. 2008)
17. QoS for Web Services: Requirement and Possible Approaches. http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/
18. Web Services Description Language (WSDL) 1.1, http://www.w3.org/TR/wsdl
19. Wei, Y.Z., Moreau, L., Jennings, N.R.: Learning users' interests by quality classification in market-based recommender systems. Knowledge and Data Engineering. IEEE Transactions. Vol.17, No. 12, 1678- 1688 (Dec. 2005)