‘Hunt ForTune’: A System for P2P E-Marketplace Matchmaking

Manish Joshi
North Maharashtra University
Department of Computer Science, Jalgaon, India
joshmanish AT gmail.com

Harold Boley
National Research Council
Institute for Information Technology Fredericton, Canada
Harold.Boley AT nrc.gc.ca

Virendra Bhavsar
University of New Brunswick
Faculty of Computer Science Fredericton, Canada
bhavsar AT unb.ca

Ravindra Vaidya
MES’s Institute of Management and Career Courses, Pune, India
rpv.imcc AT mespune.in

Abstract - Matchmaking in e-marketplaces is challenging. One of the reasons is the multifaceted nature of participants’ expectations and interest, which makes modeling of participants’ profile difficult. We formalize the multifaceted expectations and interests of participants as ‘constraints’ in those profiles. We identify and explicitly define the relevant types of constraints. We analysed several Knowledge Representation (KR) models used for developing matchmaking systems. We propose a new KR model for Web-based matchmaking systems that can represent these constraints. We describe the development of a matchmaking system that implements the proposed KR model, exemplifying its features and evaluating its performance.

Keywords - Knowledge Representation Model, Matchmaking in e-marketplaces, Multifaceted Constraints

I. INTRODUCTION
Matchmaking is considered here as the process of optimally pairing up participants from two groups, typically called sellers and buyers, according to some optimality criterion (e.g. maximum similarity). The focus of this paper is on automated matchmaking for e-marketplaces in a Web environment.

In e-marketplaces all participating sellers and buyers submit their profiles (containing descriptions of products/services offered and sought, including preferences) and wish to get a ranked list of matching profiles of other participants.

Constructing accurate profiles is a key task since system’s success depends, to a large extent, on the ability to represent participants’ interest [15]. The word ‘accurate’ here refers to how effectively all expectations of participants are modelled.

In contrast to a ‘process matchmaking’, where expectations of a process for resources (memory/processor time) are straightforward, matchmaking in e-marketplaces is complex.

Participants’ expectations can be quite numerous and multifaceted; which make them difficult to model. We propose a new KR model that captures multifaceted constraints in participant profiles, leading to the development of an effective matchmaking system.

In section 2, we discuss several aspects of matchmaking in e-marketplaces that arise due to the complex nature of participants’ expectations. This section also presents a list of matchmaking system’s features. The performance of a matchmaking system largely depends on how well it supports various types of constraints identified and explicitly defined in this section.

A Knowledge representation (KR) model is a backbone of any matchmaking system. Several KR models have been proposed for matchmaking. Each of these KR models has certain strengths and some limitations. We review some of the KR models and corresponding matchmaking systems. None of these models represents constraints of all the types discussed in section 2.

Section-3 elaborates features offered by various matchmaking systems that are based on different KR models. Details of the proposed KR model are presented in section 4. We discuss the algorithm for matchmaking in section-5. How ‘Hunt ForTune’ matchmaking system supports various features is discussed in section-6. Sample data and results generated by the system are appended as an annexure. Section-7 summarizes the evolutions. Finally, concluding remarks are given.

2. ASPECTS OF MATCHMAKING
The complex nature of participant profiles results in some interesting aspects of matchmaking. The multifaceted nature of constraints is discussed in section 2.1, whereas section 2.2 describes additional characteristics of matchmaking systems based on results generated by these systems. Three desired features of a matchmaking system are listed in section 2.3.

2.1 Multifaceted Constraints
A constraint is a condition on a profile facet (‘feature’, ‘attribute’). In the literature, mostly hard and soft constraints have been defined explicitly [17, 22]. We give below some of the possible variations of constraints. In subsequent sections we elaborate how our proposed model and matchmaking system represents all these types of constraints.

a) Hard and Soft constraints: These terms reflect the relative flexibility of participants regarding the fulfilment of a constraint. In case of a soft constraint, a participant is ready to proceed with a match even if the facet value described by his/her constraint is not satisfied by the facet value of the corresponding constraint of the counterpart profile. In contrast, a participant does not compromise with an offer/request specified as a hard constraint.

b) Range Value Constraints: The parties involved in matchmaking often provide a range for their offerings rather than a discrete value, e.g. ‘Looking for the apartment whose rent is 500 to 600’. 
This constraint should be matched with all other counterpart constraints that offer rent in the range of 500 to 600.

c) Multi-Value Constraints: Participants sometimes specify multiple discrete values as their (disjunctive) choices. For example, a constraint ‘I want a shared or a single apartment’ should be matched with all constraints offering a shared apartment as well as with all constraints offering a single apartment.

d) Preferential Constraints: For the soft constraints of a profile, a participant may wish to indicate relative preferences among various facets. For example, consider a participant’s apartment profile with rent facet preferred to facets area, type, pet-allowed. This profile can succeed in spite of low constraint satisfactions for the other facets as long as the rent constraint is highly satisfied.

e) Hidden Cost constraints: In e-business matchmaking, cost is an important facet that affects successful outcomes. Some participants (especially from the seller group) may hide facet values that could increase the cost. For example, a constraint formalizing “the rent of the apartment is $550, electricity extra”, should not succeed with the constraint of a participant who seeks a rent of $550.

2.2 Matchmaking results
The process of obtaining matchmaking results and the result (intermediate as well as final) itself characterizes a few more aspects of matchmaking.

a) Compromise match effect: A concept of soft constraints leads to the notion of a compromise match. Two constraints from two profiles have a compromise match if

i) either one or both of the constraints in comparison are soft constraints, and

ii) the values of the facets of both the corresponding constraints do not match.

In such a case, either one or both participants agree cautiously with the value mentioned in the counterpart constraint. Hence we refer to it as a ‘compromise match’.

As the compromise match is not an exact match, the similarity value should be reduced based on whether one or both participants are willing to compromise.

Different matchmaking systems have different strategies to resolve the issue of compromise matches.

b) Symmetric / Non-symmetric: If a matchmaking system returns identical results of matching a profile $P_1$ with $P_2$ and matching a profile $P_2$ with $P_1$, then the system is called a symmetric system, otherwise it is a non-symmetric system.

For example, let the profile $P_1$ have a security-deposit facet and the profile $P_2$ be without such a facet. A symmetric matchmaking system results in identical similarity values when $P_1$ is compared with $P_2$ and when $P_2$ is compared with $P_1$. In contrast, a non-symmetric matchmaking system results in different similarity values as a consequence of these comparisons.

c) Matchmaking Criterion: Based on the underlying KR model and the process (algorithm) of matchmaking, matchmaking systems use either a similarity measure or a distance measure to determine matchmaking among participants’ profiles.

d) Result Classification Categories: A participant may not be interested to have a list of all matching profiles as the result of a matchmaking process, especially when the numbers of profiles in the result is large. A participant wishes a ranked list of matching profiles preferably grouped in specific categories.

2.3 Features
Some of the important features that a matchmaking system should have are as follows.

a) Algorithm Scalability: A matchmaking system uses a particular algorithm that complements with its KR model to produce desired results. It is essential that the algorithm should handle large number of participant profiles.

b) Domain Independence: A matchmaking system that deals with the semantics of a specific domain area should be able to adapt to other domains with minimal modifications.

c) Parallelizability: With the availability of multi-core chips and high performance parallel/distributed computing, it is desirable that the algorithms used for matchmaking can be ported to suit to the distributed environment.

3. VARIOUS KR MODELS
Matchmaking systems use some KR model to represent participants’ profiles. We discuss various KR models and matchmaking systems developed using these models, in following subsections.

3.1 Array (Vector) of Features
This is a basic KR model used in early matchmaking systems. Participants’ profiles are stored either in the form of documents, a database or in a file using XML. Keywords extracted from documents are used for matchmaking among documents. A typical Information Retrieval (IR) methodology is used as the basis of matchmaking. The COINS [4] and the GRAPPA [13] matchmaking systems use this KR model.

3.2 Knowledge Representation Languages
KR languages are used to represent the concept definitions of an application domain in a structured and formally well-understood way [1]. Matchmaking systems based on KR languages emphasize the semantics, in contrast to earlier matchmaking systems which focused on the frequency of keywords. Several matchmaking systems use description logic to model domain knowledge. A semantic reasoner is used for matchmaking in some systems while other systems use customized algorithms for matchmaking. The LARKS based system [12], the Description Logic based NeoClassic Reasoner [8] and the Semantic Web language DAML-S based system [6] use KR languages to represent knowledge.

3.3 Tree
Some researchers have proposed the use of a tree structure to represent knowledge. Islam et al. [3] used a basic tree structure and proposed a matchmaking framework to identify a set of matching resources for a job, from a large collection of resources in a distributed environment. Bhavsar et al. [2] developed a matchmaking system that uses node labelled, arc labelled, arc weighted trees to represent knowledge.
3.4 Graph
Like in a tree structure, nodes and edges of a graph are used to represent concepts and relationship among these concepts. Mohaghegh et al. [7] proposed a matchmaking system in the domain of online recruitment. The IMPACT system [11] uses graph to represent knowledge.

3.5 Hybrid
A combination of different techniques is used to represent participants’ information. Ragone et al. [9] proposed a semantic matchmaking approach that integrates various knowledge representation technologies. It uses a combination of DLR-Lite, fuzzy rules, and utility theory to represent profiles of participants.

4. PROPOSED KR MODEL
We propose to represent a participant profile as a set of constraints, such that $P = \{C_1, C_2, C_3, ..., C_m\}$. Each constraint is a quadruple $C_i = <a, d, f, p>$, where $a$ is an attribute, $d$ is a set of values to describe an attribute, $f$ indicates the flexibleness of a constraint and $p$ is the priority of a constraint. All elements of a constraint are described below.

Attribute ($a$) – An attribute represents the facet. For example, if a participant has a constraint ‘need 4 bedrooms’, then the attribute of this constraint is ‘bedrooms’. This field always has an alphabetical value. Let $A$ be the domain of $a$, such that $a \in A$. An illustrative list of the set $A$ members is shown in Table 1.

Description ($d$) – Description represents a set of values assigned to the attribute of a constraint. In the example of ‘need 4 bedrooms’, the attribute ‘bedrooms’ of the constraint has the description value ‘4’. Let $D$ be the domain of $d$, $d \subset D$. $D$ contains all possible member values that a description set can have. $D$ contains all alphabetical strings that describe the attribute, or numerical values that can be assigned to an attribute, or a combination of both, or a range value having a format like $num1 num2$ such that $num1, num2 \in R$. A sample of a set $D$ is shown in Table 1.

Sometimes a participant assigns more than one value to describe the attribute of the constraint, for example, ‘looking for a shared apartment or bachelor apartment’. As the description ($d$) is a set of values, it can represent multi-value constraints. Hence for the above example, the constraint is represented as $<type, \{sharedApartment, BachelorApartment\}, f, p>$. The set of constraints ‘rent is 1500’ is available from September-1’, and ‘pets should be allowed’ can be represented as $<rent, \{1500\}, f, p>$, $<availableDate, \{Sept-01\}, f, p>$ and $<pets, \{allowed\}, f, p>$ respectively. In these examples, we have not specified any values of $f$ and $p$ for the constraints.

Consider a participant who asks for a ‘2 or 3 bedroom apartment’. In this case, the attribute ‘bedrooms’ have a description value that can be represented as a set of ‘multiple values’ or a ‘range’. Hence $<bedrooms, \{2, 3\}, Yes, p>$ and $<bedrooms, \{2–3\}, Yes, p>$ are both valid representations and have identical meanings. Figure 1 shows a rent constraint that has a range description.

<table>
<thead>
<tr>
<th>Attribute Set A</th>
<th>Description Set D</th>
</tr>
</thead>
<tbody>
<tr>
<td>area</td>
<td>downtown, riverside, north</td>
</tr>
<tr>
<td>Available Date</td>
<td>September-01</td>
</tr>
<tr>
<td>bedrooms</td>
<td>3, 1–3</td>
</tr>
<tr>
<td>cats</td>
<td>Allowed, no</td>
</tr>
<tr>
<td>dogs</td>
<td>Not-allowed</td>
</tr>
<tr>
<td>kids</td>
<td>2, No kids</td>
</tr>
<tr>
<td>laundry</td>
<td>Coin-operated, yes</td>
</tr>
<tr>
<td>pets-allowed</td>
<td>Yes, No</td>
</tr>
<tr>
<td>rent</td>
<td>500, 350, 1000–1200</td>
</tr>
<tr>
<td>smoking</td>
<td>Not Permitted, Allowed</td>
</tr>
<tr>
<td>type</td>
<td>Apartment, Shared, house</td>
</tr>
</tbody>
</table>

Flexibility ($f$) – Flexibility indicates whether the constraint is a hard or a soft constraint. $f \in F$, where $F = \{No, Yes\}$.

A ‘No’ value of $f$ (i.e. no flexibility) indicates a rigidness of the constraint, whereas value ‘Yes’ represents a soft constraint. A soft constraint is matched with any value of the counterpart as a compromise match. A constraint specification provided by a buyer as ‘house rent must be 500’ indicates a hard constraint and is represented as $<rent, \{500\}, No, p>$. A constraint description ‘Smoking is not allowed, but can smoke in balcony’, represents a soft constraint. It can be represented as $<Smoking, \{Not allowed\}, Yes, p>$.

Priority ($p$) – The priority describes the relative priority of soft constraints among other soft constraints, in a profile. The value of $p$ can be any real value grater than 0. $p \in R$. All soft constraints are initialized with the priority values of 1. The priority values for all soft constraints are set automatically to match the preferences indicated by participants.

For example, if a buyer specifies that pets allowed facet is more important to him than all remaining facets, then priority value for this constraint is set to a value grater than 1. The constraint is represented as $<pets, \{allowed\}, No, 1.1>$, and all remaining constraints will have $p$ values as 1. Note that, the value of flexibility in this example, is ‘No’, indicating a hard constraint. These priority values ultimately used to rank the service represented by the facet.

The Figures 1 and 2 illustrate how a buyer’s (a tenant’s) profile and a seller’s (a computer owner’s) profile can be represented in our model. The description of the participants profiles is followed by a node representation (Figures 1(a), 2(a)) and a quadruple representation (Figures 1(b), 2(b)).

Profile-1 – Tenant (Buyer)
I am a mature student looking for an affordable shared or single apartment on the south side of Fredericton for September. Finishing up my last year at UNB, I smoke but can adjust with non-smoking apartment. rent – 400 to 450. Please contact if anything is available, thanks!
5. ALGORITHM

The similarity value between any two profiles is defined as a function of attribute, description, flexibility and priority values of all constraints from both profiles. For any two profiles $P_x$ and $P_y$, where $P_x$ has $m$ constraints and $P_y$ has $n$ constraints, a similarity value is given by,

$$Sim(P_x, P_y) = \prod_{i=1}^{m} S(C_i, C_j)$$

where the function $S(C_i, C_j)$ calculates an intermediate similarity value using steps given in the algorithm below.

The attribute, description, flexibility and priority values of a constraint are accessed using a notation $C.a$ which means the attribute value of the constraint $i$.

Algorithm

1: if $(C_i.a = C_j.a)$ then
2:   if $(C_i.d = C_j.d)$ then
3:     return $S(C_i, C_j) = C_i.p \times C_j.p$
4:   else
5:     return $S(C_i, C_j) = \beta$
6:   elseif $(C_i.f = Yes) \text{ AND } (C_j.f = Yes)$ then
7:     return $S(C_i, C_j) = C_i.p \times C_j.p \times \beta$
8:   else
9:     return $S(C_i, C_j) = Omission \text{ Penalty}$
10: end
11: end
12: if $(C_i.a < C_j.a)$ then
13:   move on to next $C_i$ and $C_j$
14: else
15:   move on to next $C_j$
16: end
17: end

The algorithm compares two constraints of two profiles. If the attributes of both the constraints are same then an intermediate similarity value is calculated by checking the description values. If the description values are not same then an intermediate similarity value is calculated by considering the flexibility of the constraints. When hard constraints in two profiles do not match, instead of reducing a similarity value immediately to zero, we compute relative difference between the two corresponding description values of these attributes. A routine relativeDifference computes relative difference, which is later used to calculate a similarity value. Note that for numeric and alphabetical values of $d$, separate routines are required to obtain relative differences.

We make sure that an intermediate similarity value for such constraints is reduced substantially.

Numeric relative difference between profiles having rent as 500 and 700 (where numeric difference is 200) is not the same as profiles having rent as 1500 and 1700. Rather the first difference (i.e. between 500 and 700) is relatively greater than the second.
The parameters $\alpha$ and $\beta$ are 
*compromise count factors* used in case of compromise match and its usage is elaborated in next section.

### 6. ‘HUNT FORTUNE’ FEATURES

In the previous section, it is shown how the proposed model represents multifaceted constraints. In this section, we describe additional features supported by the Hunt ForTune matchmaking system that is based on the proposed KR model.

**Preferential Constraints:** Our framework facilitates participants to indicate the relative importance among soft constraints, if any. For example, a participant can indicate $\text{facets}_1 > \text{facets}_3 > \text{facets}_2$, using an interface and appropriate priority values are assigned to the corresponding constraints. Figure 3 shows screenshot of the GUI for a matchmaking system developed by us.

Each constraint is initialized with priority value 1 and it is gradually incremented after participant clicks on ‘+’ button placed beside the priority value of a facet (see Figure 3). This interface allows participant to input his/her constraints. Using this interface the participant can indicate preferences among facets easily. In Figure 3, the preference of an ‘availableDate’ facet is set to 1.1, while for all other soft constraints the priority is set to 1.

![Figure 3. Screenshot of the GUI for profile entry](image)

**Hidden Cost Constraints** – We propose that a profile with a hidden cost constraint should be penalized in the process of matchmaking. Hence a constraint, which carries hidden cost, has to bear the hidden cost penalty.

In our matchmaking system, we reduce the priority value of the hidden cost constraint to 0.9. This value is less than priority values of all remaining constraints (all other constraints have priority values of at least 1).

Due to the penalty, in terms of reduction in priority, similarity value of a profile that contains hidden cost constraint, will be less than a profile that do not have hidden cost constraint.

**Symmetry/Non-symmetry:** We introduce a parameter *omission penalty*, and user can set its value. This parameter value is used to reduce the resulting similarity value while matchmaking, for each constraint that is present in the Seller’s profile but missing from the Buyer’s profile; or vice versa.

If the value of omission penalty is set to 0, the system shows characteristics of a symmetric matchmaking system, i.e. $\text{Sim}(P, P) = \text{Sim}(P, P)$. For any other value of omission penalty such that $0 < \text{omission penalty} \leq 1$, the matchmaking system exhibits non-symmetric characteristics from buyers and sellers points of views.

**Compromise Match Effect:** A compromise match is not an exact match, hence a similarity value between corresponding profiles should be reduced. In our matchmaking system, when there is a compromise match between two constraints, an intermediate similarity value (given by the function $S$ in equation 1) is reduced by a certain factor. Consider an example of a soft constraint by a seller, “Prefer a non-smoker but ready to deal with a smoker” and a buyer’s soft constraint as “I am looking an apartment where smoking is allowed but ready to rent a non-smoking apartment too”. These two constraints have a compromise match. As both of the participants are ready to compromise with their preferred choices, it is likely that these two participants can reach an agreement. Hence a similarity value in case of a compromise match is influenced by the count (compromise count) of participants (one or both) willing to compromise.

We propose two *compromise count factors*, $\alpha$ and $\beta$ to reduce a similarity value, in case of a compromise match. The values of $\alpha$ and $\beta$ are set to less than 1. An intermediate similarity value is multiplied by these factors to obtain an expected reduction in a similarity value.

If a compromise count is one, then there are relatively less chances of an agreement as only one participant is ready to compromise. The factor $\alpha$ represents this case, while the factor $\beta$ is used when compromise count is two.

We set the values of $\alpha$ and $\beta$ such that a higher similarity value shall be resulted for a compromise match where both participants are ready to compromise and a lower similarity value shall be resulted if only one participant is ready to compromise.

**Scalability** – The KR model uses simple set of nodes to capture key information associated with the participant profiles. It avoids overhead of building complex data structures like graph and tree. The algorithm compares two profiles and generates the similarity value in linear time. Hence we could expect this approach to generate results in satisfactory amount of time even for large number of profiles. The algorithm for matchmaking can easily be converted to suit for a distributed/parallel computing.

**Domain Independence** – The KR model is totally independent of domain and can be applicable to many domains. This KR model describes a general technique to capture essence of any type of constraint. It has the provision to capture various options offered/demanded by participant in any constraint.

In order to be useful in any domain a specific ontology for that domain shall be required. The semantic relative difference routine used in the algorithm and other features largely depends upon domain knowledge.

**Automatic categorization** – As the nodes are created by considering attribute values and description values of constraints among profiles, the KR model can be programmed to count and categorize profiles based on these values. A more descriptive categorization shall be available after processing of all profiles.
7. EVALUATION

We have obtained results of the matchmaking system developed using our KR model for a house rental domain. Our system supports all the types of constraints discussed in section 2.1. The system generates an appropriate list of similarities among profiles. The system facilitates users to determine the ranking of matching profiles by tuning the values of parameters like the omission penalty and the compromise count factors.

8. CONCLUSION

We have proposed a new knowledge representation model to represent participants’ complex constraints in an automated matchmaking. We have reviewed several matchmaking systems developed using different KR models. We analysed these systems based on various features discussed in the paper. The features offered by the ‘Hunt Fortune’ matchmaking system developed using our proposed KR model are discussed. We showed that our system offers many additional features as compared to other matchmaking systems. A further analysis of the effect of changes in the parameter values on the ranking of matching profiles is underway.

9. REFERENCES


APPENDIX-1

Each of the profiles P-1 to P-6 is matched with profiles P-8 to P-14 to obtain similarity values. Only those matching profiles are displayed in the result where similarity value is non-zero.
<table>
<thead>
<tr>
<th>House Owner’s Profile</th>
<th>Buyer – Tenant’s Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>P-1</td>
<td>P-8</td>
</tr>
<tr>
<td>&lt;bedrooms, {4}, No, 1&gt;</td>
<td>&lt;available, {Sept-1}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;laundry, {yes}, No, 1&gt;</td>
<td>&lt;bedrooms, {1}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;lease, {1-year}, No, 1&gt;</td>
<td>&lt;rent, {100-400}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;rent, {1700}, No, 1&gt;</td>
<td>&lt;type, {apartment}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;type, {apartment}, No, 1&gt;</td>
<td></td>
</tr>
<tr>
<td>P-2</td>
<td>P-9</td>
</tr>
<tr>
<td>&lt;available, {Sept-1}, No, 1&gt;</td>
<td>&lt;available, {Sept-1}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;pets, {no}, No, 1&gt;</td>
<td>&lt;rent, {375}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;rent, {395}, No, 1&gt;</td>
<td>&lt;type, {bachelor, room}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;smoke, {no}, No, 1&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;type, {bachelor, No}, 1&gt;</td>
<td></td>
</tr>
<tr>
<td>P-3</td>
<td>P-11</td>
</tr>
<tr>
<td>&lt;available, {Sept-1}, No, 1&gt;</td>
<td>&lt;bedrooms, {2}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;bedrooms, {3}, No, 1&gt;</td>
<td>&lt;kids, {yes}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;rent, {600-900}, No, 1&gt;</td>
<td>&lt;rent, {500}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;security, {700}, No, 1&gt;</td>
<td>&lt;type, {apartment}, Yes, 1&gt;</td>
</tr>
<tr>
<td>&lt;type, {apartment}, No, 1&gt;</td>
<td></td>
</tr>
<tr>
<td>P-4</td>
<td>P-12</td>
</tr>
<tr>
<td>&lt;bedrooms, {1}, No, 1&gt;</td>
<td>&lt;available, {Sept-1}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;parking, {1}, No, 1&gt;</td>
<td>&lt;parking, {1}, Yes, 1&gt;</td>
</tr>
<tr>
<td>&lt;rent, {625}, No, 1&gt;</td>
<td>&lt;rent, {500}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;type, {apartment}, No, 1&gt;</td>
<td>&lt;type, {bachelor}, No, 1&gt;</td>
</tr>
<tr>
<td>P-5</td>
<td>P-13</td>
</tr>
<tr>
<td>&lt;rent, {300}, No, 1&gt;</td>
<td>&lt;pets, {yes}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;type, {room}, No, 0.99&gt;</td>
<td>&lt;rent, {0}, Yes, 1&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;type, {room}, Yes, 1&gt;</td>
</tr>
<tr>
<td>P-6</td>
<td>P-14</td>
</tr>
<tr>
<td>&lt;available, {Aug-1}, No, 1&gt;</td>
<td>&lt;area, {downtown}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;bedrooms, {2}, No, 1&gt;</td>
<td>&lt;available, {Sept-1}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;laundry, {yes}, No, 1&gt;</td>
<td>&lt;bedrooms, {2}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;parking, {2}, No, 1&gt;</td>
<td>&lt;kids, {no}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;rent, {900}, No, 1&gt;</td>
<td>&lt;laundry, {yes}, No, 1&gt;</td>
</tr>
<tr>
<td>&lt;type, {apartment}, No, 1&gt;</td>
<td>&lt;pets, {yes}, No, 1&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;rent, {800}, Yes, 1&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;type, {apartment}, No, 1&gt;</td>
</tr>
</tbody>
</table>

**Matchmaking Results**

Similarity value - profile 1 Vs. profile 13 is -->0.9412
Similarity value - profile 1 Vs. profile 14 is -->0.397635
Similarity value - profile 1 Vs. profile 11 is -->0.1396
Similarity value - profile 2 Vs. profile 12 is -->0.985
Similarity value - profile 2 Vs. profile 8 is -->0.9652
Similarity value - profile 3 Vs. profile 14 is -->0.4315
Similarity value - profile 4 Vs. profile 14 is -->0.9506
Similarity value - profile 4 Vs. profile 13 is -->0.946
Similarity value - profile 5 Vs. profile 11 is -->0.4703
Similarity value - profile 5 Vs. profile 9 is -->0.995
Similarity value - profile 5 Vs. profile 13 is -->0.9751
Similarity value - profile 5 Vs. profile 8 is -->0.9702
Similarity value - profile 5 Vs. profile 11 is -->0.9653
Similarity value - profile 6 Vs. profile 13 is -->0.93639
Similarity value - profile 6 Vs. profile 11 is -->0.4268
### Table-2: Cross dimensional analysis of various KR models used for matchmaking systems

<table>
<thead>
<tr>
<th>KR Model</th>
<th>Matchmaking System</th>
<th>Multifaceted Constraints</th>
<th>Result Classification Categories</th>
<th>Algorithm Details</th>
<th>Algorithm Process</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hard / Soft</td>
<td>Range Value</td>
<td>Preferential Value</td>
<td>Alternate Value</td>
<td>Hidden Cost</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
Table-2: Cross dimensional analysis of various KR models used for matchmaking systems (Continue…)

<table>
<thead>
<tr>
<th>KR Model</th>
<th>Matchmaking System</th>
<th>Multifaceted Constraints</th>
<th>Result Classification Categories</th>
<th>Algorithm Details</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Hard / Soft</td>
<td>Range Value</td>
<td>Preferential</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>Tree</td>
<td>A Weighted-Tree (2004) [3]</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Hybrid</td>
<td>Vague Knowledge Bases for Matchmaking in P2P E-</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Combination of Description Languages,</td>
<td>Marketplaces (2007) [16]</td>
<td></td>
<td>Discrete</td>
<td></td>
</tr>
<tr>
<td>Fuzzy Rules and Utility theory</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proposed (set of Nodes)</td>
<td>(2009)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>