Knowledge Representation in Matchmaking Applications
Joshi, Manish; Bhavsar, Virendrakumar C.; Boley, Harold

Publisher's version / Version de l'éditeur:

Advanced Knowledge-Based Systems: Models, Applications &Research, 1, pp. 29-49, 2010-01-01

Chapter

Knowledge Representation in Matchmaking Applications

***Manish Joshi*[1], *Virendrakumar C. Bhavsar*[2], *Harold Boley*[3]**

Absract Matchmaking systems' features and success largely depends on how effectively participants' product/service (profile) descriptions are modelled. We formalize the multifaceted expectations and interests of participants as 'constraints' in those profiles. We identify and define the relevant types of constraints and explicitly document challenges in matchmaking.
Knowledge Representation Model (KRM) determines how different types of constraints are represented in any matchmaking system. We analyze role of KRM in matchmaking systems by reviewing seven different KRMs and listing features offered by matchmaking systems that use these KRMs. We propose a new KRM that represent various types of constraints. We describe the development of a matchmaking system that implements the proposed KRM, exemplifying its features and evaluating its performance.

1. INTRODUCTION

With the advancement of Internet technology and rapid growth in online trading, numbers of service providers are showcasing their websites using which users (sellers as well as buyers) can trade. These websites are becoming new virtual marketplaces. 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. 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). Automated matchmaking is a topic of research for several years. It has extended to electronic negotiations, auctions, bartering etc. The focus of this chapter will be on automated matchmaking for e-marketplaces in a Web environment.

[1] Department of Computer Science, North Maharashtra University, Jalgaon, MS, India.
joshmanish@gmail.com
[2] Faculty of Computer Science, University of New Brunswick, Fredericton, NB, Canada.
bhavsar@unb.ca
[3] National Research Council, Institute for Information Technology Fredericton, Fredericton, Canada.
Harold.Boley@nrc.gc.ca

Constructing accurate profiles is a key task since matchmaking system's success depends, to a large extent, on the ability to represent participants' interest [16]. The word 'accurate' here refers to how effectively all expectations of participants are modelled. Hence, knowledge representation model plays an important role in all matchmaking applications. Participant's expectations, also called *constraints*, are relatively numerous and multifaceted which make it difficult to model. In contrast to a 'process matchmaking', where expectations of the process for resources (memory/processor time) are straightforward, matchmaking in e-marketplaces is complex.
We identify and explicitly define various forms that constraints can take. We listed several different types of constraints that a matchmaking system should support. The ability of a matchmaking system to support all types of constraints determines its performance. In other words we present criteria to measure performance of any matchmaking system. We discuss in detail complex nature of participants' profiles and expectations as the challenging aspects of matchmaking.
A knowledge representation model is a backbone of any matchmaking system. Several KRMs have been proposed for matchmaking. Each of these KRMs has certain strengths and some limitations. We review some of the KRMs and corresponding matchmaking systems. We compare these matchmaking systems based on various aspects of matchmaking. In particular we discuss the features offered, technique used and matching process (algorithm) of the matchmaking systems that use *Array of Features*, *Knowledge Representation Language*, *Database*, *Tree*, *Graph*, and *Hybrid* KRMs. We provide a tabular comparative study of all features offered by matchmaking systems that are based on these KRMs.
We propose a new knowledge representation model for a Web-based matchmaking environment, which can represent all types of constraints defined in participants' profile. We discuss this model in detail. We develop a matchmaking system implementing the proposed KRM to exemplify its features and to evaluate its performance. We shall discuss our matchmaking system in detail.
In short the objectives of the chapter are to elaborate importance and usage of KRMs in matchmaking. We also want to discuss the general criteria for a matchmaking system to be more effective.

2. CHALLENGES IN MATCHMAKING

The complex nature of participant profiles results in some interesting and challenging aspects of matchmaking. Participant constraints are numerous and can be of different forms. Each constraint needs special treatment. Participant participating in matchmaking has certain expectations regarding the results. The matchmaking systems are expected to demonstrate specific features.

2.1. Types of Constraints

A constraint is a condition on a profile facet ('feature', 'attribute'). In the literature, mostly *har*d and *soft* constraints have been defined explicitly [18, 20, 23]. 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 (disjunctive) as their 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 users 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 users are willing to compromise.
Different matchmaking systems have different strategies to resolve the issue of compromise match.

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) 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 are large. A participant wishes a ranked list of matching profiles preferably grouped in specific categories.

2.3. Algorithm Scalability

A matchmaking system uses a particular algorithm that complements with its KRM to produce desired results. It is essential that the algorithm should scale reasonably (in terms of computational complexity) to handle large number of participant profiles.

2.4. 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. A matchmaking system should easily be plugged-in with other domain.

2.5. 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.

Let's analyze various KRMs and their corresponding matchmaking applications in the next section by discussing features offered by these systems.

3. ANALYSIS OF VARIOUS KRMs

Many matchmaking systems are available. A matchmaking system uses some KRM to represent participant profiles. We discuss various KRMs and the matchmaking systems developed using them in the following subsections.

3.1. Array (Vector) of Features

This is a basic knowledge representation model used in early matchmaking systems. User profiles are stored either in the form of document or in a database or in a file using XML. The keywords extracted from the documents are used for matchmaking among the documents. A typical Information Retrieval (IR) methodology is used as the basis of matchmaking. Let's briefly describe structure of two matchmaking systems.

a) COINS system
Kuokka and Harada [12] presented one of the early matchmaking systems, COINS (COmmon INterest Seeker). It uses a distance measure of information retrieval technique to carry out matchmaking on free text. The COINS system converts the free text document into a document vector, which is used later for processing. It uses SMART [19] information retrieval system to process and match free text and document vectors.

b) GRAPPA system
Veit et al. [23] have developed the Generic Request Architecture for Passive Provider Agent (GRAPPA) matchmaking framework and library system in 2001. The matchmaking engine accepts a

set of offers and requests as an input. A distance function recursively computes the distance values (between 0 and 1) for different profile subtypes. Based on these distance values, the system returns a ranked list of the best ‘k’ candidate profiles matching to a given profile. The system used a typical vector space model to identify distances among the keywords occurring in a document. The system maintains a document vector and calculates a term-frequency-inverse-document-frequency factor (tf-idf factor), which is used in further processing to determine the similarity between two documents. The GRAPPA system uses XML to describe user profiles.

3.2. Database

A matchmaking system that uses a database to represent knowledge is developed for a specific domain. The domain specific information is organised appropriately in the database. Matchmaking in this case basically relies on IR based techniques. The ontological part is embodied with the system to obtain semantically acceptable results of matchmaking.

Liesbeth et al. [14] developed a matchmaking system to respond to a learner’s request by matching profiles of other learners who wish to share knowledge, by determining their content competence, sharing competence, eligibility and availability.

The database system is used to store, learning contents that are organized in courses and user (learner) profiles. A user profile consists of completed courses, current courses, activities, calendar and other information.

A leaner input a query to the system using the request module interface and the query data is stored in the database. A Latent Semantic Analyser (LSA) maps the content question on the available documents in the database to generate a list of all suitable matching resources, i.e. other learners who are content competent, sharing competent, eligible, available, etc.

3.3. Tree

Some researchers have proposed the use of a tree structure to represent the knowledge. A basic tree structure is used by Islam et al. [10]. They proposed a matchmaking framework to identify the set of matching resources for a job, from a large collection of resources in a distributed environment.

Bhavsar et al. [3] developed a matchmaking system that uses node labelled, arc labelled, arc weighted trees to represent knowledge. Nodes of a tree are used to represent the concept and branches represent the relationship. A ‘weight’ is assigned to an arc to signify the importance of corresponding relationship in matchmaking. A recursive bottom up approach based customized algorithm is designed to evaluate similarity among such trees and list of matching profiles is generated.

3.4. Graph

Like in a tree structure, the nodes and edges of a graph are used to represent concepts and relationship among these concepts. Mohaghegh et al. [15] proposed a matchmaking system in the domain of online recruitment. It compares available resumes with a special kind of skills ontology in which different skills and relationship among skills are represented as nodes and edges of a graph. Similarities among skills are described with the help of a graph structure. Advertisements and resumes are attached to appropriate nodes of the graph based on the contents of documents. A path-set is used to evaluate score between a given advertisement and a resume. A specific function calculates score values between an advertisement and all resumes. Ranking of resumes is provided as a result of the matchmaking.

In the IMPACT system [21], Yellow Pages Servers play the role of matchmaking agents. Offers and requests are described in a simple data structure that represents a service by a verb and one or two nouns. The matchmaking process finds k-nearest service names, and agents who provide these

services. All these service names are within a specified distance d. The weights on edge of the graph reflect distances between a set of verbs and nouns, which are represented as nodes of a graph.
Bellur and Kulkarni [2] used a variation of graph, *Bipartite graph*, for matchmaking of web services.

3.5. Knowledge Representation Languages

Knowledge Representation (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 that focus on the frequency of keywords. Several matchmaking systems are developed that use description logic to model the domain knowledge. A semantic reasoner is used for matchmaking in some of the systems. In other systems customised algorithms have been developed for matchmaking.
Some of the matchmaking systems that use knowledge representation languages are described below.

a) LARKS based system
Sycara et al. [22] have proposed an agent capability description language called LARKS (Language for Advertisement and Request for Knowledge Sharing), that is used in a Retsina multi-agent infrastructure framework. A matchmaking process is carried out at five possible levels, namely, context matching, profile comparison, similarity matching, signature matching and semantic matching.
A standard technique of the Information Retrieval is used for the syntactic matching which includes profile comparison, similarity matching, and signature matching. Whereas, the semantic matchmaking is achieved using a local ontology, written in a specific concept language ITL.
The system identifies three types of matches among profiles. An exact match is the most accurate type of match, plug-in match is a less accurate, and relaxed match is the least accurate type of match [22].

b) Description Logic based NeoClassic Reasoner
Di Noia et al. [5] developed a system that facilitates semantics-based matchmaking. It uses a description logic (DL) based framework to represent knowledge. They have proposed a three-way classification of matches as - exact, potential and partial. The system provides a numeric score that indicates how far the demand is with reference to the supplies. These numeric scores are used to rank the results. An apartment-rental ontology based case study is discussed in the paper.

c) Semantic Web language DAML-S based System
Li and Horrocks [13] developed a service matchmaking system that uses DAML-S service description language and ontology. DAML-S uses a set of mark-up language constructs to describe the properties and capabilities to represents web services in unambiguous, computer-interpretable form. A prototype matchmaking that uses a DL reasoner (RACER) [8] is developed. The prototype system matches service advertisements and requests using ontology based service descriptions semantics. The system proposes the concept of *degree of match* which is used to classify the match results in five different classes. The query is divided into volatile query and persistent query based on the duration for which it remains active.
One of the earliest matchmaking systems proposed by Finin et al. [6] was based on the KQML and used a Rule Based Approach for matchmaking.
Hoffner et al. [9] proposes matchmaking as a starting point of negotiations between a demand and a supply in a peer-to-peer way. The matchmaking engine (MME) handles supplies/demands as properties and rules. The properties are name-value pairs constructed using an extension of the Corba Trading service language. Rules are constructed using a generic script language. The matching is accomplished by comparing properties and verifying rules.
Based on Semantic Web concepts and ontologies developed, Gupta et al. [7] claim to improve the performance of web service query matchmaking.

These all matchmaking systems use knowledge representation languages to represent user's profile.

3.6. Hybrid

A combination of different techniques is used to represent user information. Ragone et al. [18] propose a semantic matchmaking approach that mixes various knowledge representation technologies. It uses a combination of DLR-Lite, fuzzy rules, and utility theory to represent users profiles. In particular fuzzy rules are used to represent the concept of hard *constraints* and *soft* constraints. Sellers and buyers can assign utility values to indicate preferences among the constraints. The system uses vague Datalog rules that can assign appropriate scores depending upon the values in the profiles so that fuzzy descriptions like 'cheap cars' can be incorporated. The system returns top k matches and ranks them based on the score which is computed using the datalog rules, utility values assigned to the constraints, etc.

3.7. Other

A 'one-input transition system' based model, slightly similar to Deterministic Finite Automaton (DFA), proposed by Çelebi et al. [4] and a Neural Network [24] are also used as KR models for 'process matchmaking'. As mentioned earlier, we do not explore these KRM based systems as we wish to compare matchmaking systems used in e-marketplaces.
A fuzzy linguistic approach is used to model and match buyer's preference with products [17]. But for the process of matchmaking the system considers only two features of the products which make the profile far simple.
All matchmaking systems based on the principles of different KRMs, are compared using some of the characteristics listed in earlier section. Table 2 at the end of paper shows the cross-dimensional analysis of such systems.

4. PROPOSED KRM

We propose to represent a participant profile as a set of constraints, such that $\mathrm{P} = \{C_1, C_2, C_3, ... C_m\}$. 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 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 $num1 \cdots num2$ such that $num1, num2 \in R$. A sample of a set D is shown in Table 1.

Sometimes a party 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', '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 user 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, { $2 \cdots 3$ }, Yes, p> are both valid representations and have identical meanings. Figure 1 shows a rent constraint that has a range description.

Table 1: A example Attribute (A) and Description (D) sets.

Attribute Set A	**Description Set D**
Area	downtown, riverside, north
Available	September-01, Fall, Summer
Bedrooms	3, $1 \cdots 3$
Cats	Allowed, no
dogs	Not-allowed
Kids	2, No-kids
Laundry	Coin-operated, yes
Pets	Yes, Mo
rent	500, 350, $1000 \cdots 1200$
Smoking	Not-Permitted, Allowed
Type	Apartment, Shared, house

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)

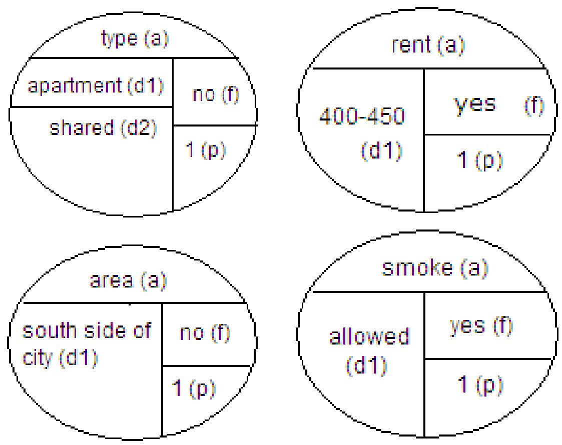
The *priority* describes the relative priority of soft constraints among other soft constraints, in a profile. The value of p can be any real value grater than 0. $p \in R$. All soft constraints are initialized with the priority values of 1. The priority values for all soft constraints are set automatically 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!



(a) Constraints as nodes

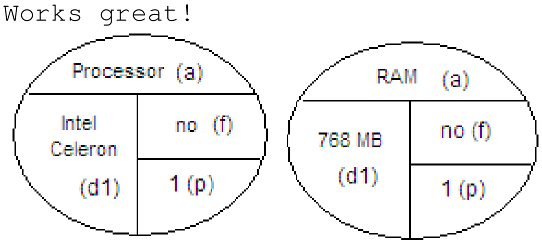
<type, {apartment, shared}, No, 1>
<rent, { 400···450 }, Yes, 1>
<area, {South side}, No, 1>
<smoke, {allowed}, Yes, 1>
<available, {Sept-01}, No, 1>

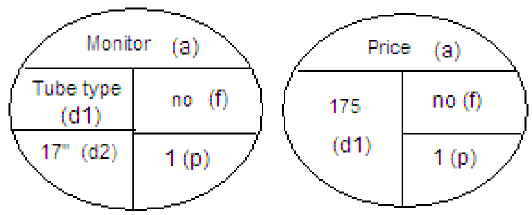
(b) Constraints as quadruples

Figure 1. Representation of the constraints of a Buyer

Profile-2 – Computer Owner (Seller)

Intel Celeron processor at 1.7 Ghz, 768 MB RAM, 40 GB hard drive, Nvidia GeForce FX5700LE 256 mb video card, floppy, CD burner, 17" tube type monitor, Windows XP Pro installed (no disk). Completely configured and ready for high speed internet connection, includes AVG anti-virus. Works great!





(a) Constraints as nodes

```
<Processor, {Intel Celeron}, No, 1>
<Speed, {1.7 GHz}, No, 1>
<RAM, {768 MB}, No, 1>
<HDD, {40 GB}, No), 1>
<VideoCard, {256 MB}, No, 1>
<OpticalDevice, {CD Writer}, No, 1>
<Monitor, {17 inch, tube type}, No, 1>
<OS, {Windows XP}, No, 1>
<Internet, {high speed}, No, 1>
<Softwares, {AVG antivirus}, No, 1>
<Price, {175}, No, 1>
<Type, {personal computer}, No, 1>
```

(b) Constraints as quadruples

Figure 2. Representation of the constraints of a Seller

Appendix-1 shows sample Seller's, Buyer's profiles and matchmaking results obtained for these sample profiles.

The next section elaborates algorithmic steps to compute similarity value between any two given profiles.

5. MATCHMAKING 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,

$$Sim(\mathrm{P}_x, \mathrm{P}_y) = \prod_{\text{i=1 to m, j=1 to n}} S(C_i, C_j) \tag{1}$$

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 $C_i.a$ which means the attribute value of the constraint i.

```
if (Ci.a=Cj.a) then
     if (Ci.d=Cj.d)then
```

```
            return S(Ci , Cj)=  Ci .p  ×  Cj.p
        else
            if (Ci.f=No) AND (Cj.f = No) then
               return S(Ci,Cj)= Ci.p × Cj.p ×
               relativeDifference(Ci.d ,Cj.d)
            elseif (Ci.f=Yes) AND (Cj.f = Yes)
               return S(Ci,Cj)= Ci.p × Cj.p ×β
            else
               return S(Ci,Cj)= Ci.p × Cj.p ×α
        move on to next Ci and Cj
    if (Ci.a < Cj.a) then
            return S(Ci,Cj)= Omission Penalty
            move on to next Ci
    if (Ci.a > Cj.a) then
            return S(Ci,Cj)= Omission Penalty
            move on to next Cj
```

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 α and β are *compromise count factors* used in case of compromise match and its usage is elaborated in next section.

Appendix-2 shows an example of how matchmaking algorithms results appropriate similarity values when profiles are matched with each other.

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 KRM.

6.1. Preferential Constraints

Our model facilitates participants to indicate the relative importance among soft constraints, if any. For example, a participant can indicate $facet_1 > facet_5 > facet_3$ using an interface and appropriate priority values are assigned to the corresponding constraints. Figure 3 shows screenshot of the GUI of the 'Hunt ForTune' matchmaking system.

Each constraint is initialized with priority value 1 and it is gradually incremented after user 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 soft 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.

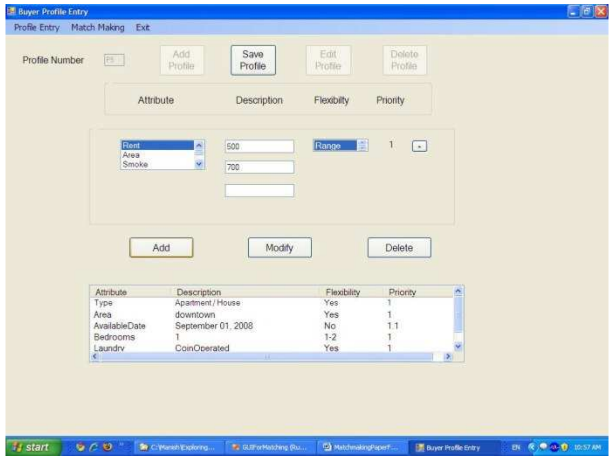


Figure 3. Screenshot of the GUI for profile entry

6.2 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 the priority values of all remaining constraints (all other constraints have priority values of at least 1).

Due to the penalty, in term 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.

6.3 Symmetry/Non-symmetry

We introduce a parameter *omission penalty*, and its value can be set by a user. 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 an omission penalty is set to 0, the system shows characteristics of a symmetric matchmaking system, i.e. $Sim(\mathrm{P}_x,\mathrm{P}_y) = Sim(\mathrm{P}_y,\mathrm{P}_x)$. For any other value of omission penalty such that $0 < \text{omission penalty} \leq 1$, the matchmaking system exhibits non-symmetric characteristics from buyers or sellers viewpoint.

6.4 Compromise Match Effect

As a compromise match is not an exact match, 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*, α and β to reduce a similarity value, in case of a compromise match. The values of α and β 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 α represents this case, while the factor β is used when compromise count is two.

We set the values of α and β 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.

6.5 Result Classification Categories

A user desires to obtain a list of matching profiles classified among categories and ranked within the categories.

We propose following six categories for matchmaking results.

1. Matching all hard constraints and matching all soft constraints.
2. Matching all hard constraints and matching some soft constraints and absence of remaining soft constraints in counterpart profile (leading to further action like – inquiring).
3. Matching all hard constraints and absence of all soft constraints in counterpart profile.
4. Matching all hard constraints, some compromise match, and some missing constraints.
   A. Compromise match constraints where both parties willing to compromise.
   B. Compromise match constraints where only one party is willing to compromise.
5. Not matching hard constraints and the margin of difference in description values is less.
6. Not matching hard constraints and the margin of difference in description values is high.

6.6 Scalability

The KRM 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.

6.7 Domain Independence

The KRM is totally independent of domain and can be applicable to many domains. This KRM 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.

6.8 Automatic categorization

As the nodes are created by considering attribute values and description values of constraints among profiles, the KRM 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 KRM for a house rental domain. Our system supports all the types of constraints discussed in ‘Challenges in Matchmaking’ section. 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. The detailed study of change in parameter values on matchmaking result classification is in progress.

8. CONCLUSION

We discuss role of KRM in automated matchmaking. We enlist several features that matchmaking systems should exhibit. We used these features to review several KRMs and their corresponding matchmaking systems.

We have proposed a new model for knowledge representation that represents complex constraints of users participating in automated matchmaking. We discuss how our system offers many additional features as compared to other matchmaking systems.

ACKNOWLEDGEMENTS

This research work was partially supported by a post-doctoral fellowship from ACEnet, Canada (www.ace-net.ca) and a research grant of Dr. Bhavsar from the Natural Sciences and Engineering Research Council of Canada (NSERC). The research work was partly carried out when Dr. Bhavsar was visiting the International Institute of Information Technology, Pune, India.

REFERENCES

[1] F. Baader, D. Calvanese, D. Mcguinness et al., “ The Description Logic Handbook: Theory, Implementation and Applications,” Cambridge University Press, Cambridge, MA, 2003.

[2] U. Bellur and R. Kulkarni, "Improved matchmaking algorithm for semantic web services based on bipartite graph matching," in *IEEE International Conference on Web Services,* 2007,

[3] V. C. Bhavsar, H. Boley and Y. Lu, "A Weighted-Tree Similarity Algorithm for Multi-Agent Systems in e-Business Environments," *Computational Intelligence,* vol. 20, pp. 584-602, 2004.

[4] R. Çelebi, H. Ellezer, C. Baylam, I. Cereci and H. Kılıç, "Process Matchmaking on a P2P Environment," *Procceding of International Conference on Web Intelligence and Intelligent Agent Technology,* pp. 463-466, 2006.

[5] T. Di Noia, E. Di Sciascio, F. M. Donini and M. Mongiello, "A System for Principled Matchmaking in an Electronic Marketplace," *International Journal of Electronic Commerce,* vol. 8, pp. 9-37, 2004.

[6] T. Finin, R. Fritzson, D. Mckay and R. McEntire, "KQML as an agent communication language." in *Third International Conference on Information and Knowledge Management,* 1994, pp. 456-463.

[7] C. Gupta, R. Bhowmik, R. H. Michael, M. Govindaraju and W. Meng, "Improving performance of web services query matchmaking with automated knowledge acquisition," in *International Conference on Web Intelligence,* 2007, pp. 559-563.

[8] V. Haarslev and R. Moller, "RACER System Description," *Proceeding of the International Joint Conference on Automated Reasoning (IJCAR 2001) Lecture Notes in Artificial Intelligence,* vol. 2083, pp. 701-705, 2001.

[9] Y. Hoffner, A. Schade, C. Facciorusso and S. Field, "Negotiation Protocol Characterisation and Mechanism for Virtual Markets and Enterprises" *Collaborative Business Ecosystems and Virtual Enterprises,* 2002.

[10] M. R. Islam, M. Z. Islam and L. Nazia, "A Tree-based Approach to Matchmaking Algorithms for Resource Discovery," *International Journal of Network Management,* 2008.

[11] M. Joshi, V. Bhavsar and H. Boley, "A Knowledge Representation Model for Match-Making Systems in e-Marketplaces," Proc. of the 11th International Conference on e-Commerce (ICEC 2009), August 12-15, 2009, Taipei, Taiwan, pp. 362-365, 2009.

[12] D. Kuokka and L. Harada, "Integrating Information via Matchmaking," *Journal of Intelligent Information Systems,* vol. 6, pp. 261-279, 1996.

[13] L. Li and I. Horrocks, "A Software Framework for Matchmaking Based on Semantic Web Technology," *International Journal of Electronic Commerce,* vol. 8, pp. 39-60, 2004.

[14] K. Liesbeth, P. Rosmalen, P. Sloep, F. Brouns, M. Koné and R. Koper, "Matchmaking in Learning Networks: Bringing Learners Together for Knowledge Sharing," *The Netherlands Interactive Learning Environments,* vol. 15, pp. 117-126, 2007.

[15] S. Mohaghegh and M. R. Razzazi, "An Ontology Driven Matchmaking Process," *World Automation Congress,* vol. 16, pp. 248-253, 28 June - 1 July 2004. 2004.

[16] M. Montaner, B. Lopez and Josep LLuis, De La Rosa, "A Taxonomy of Recommender Agents on the Internet," *Artificial Intelligence Review,* vol. 19, pp. 285-330, 2003.

[17] A. C. Ojha and S. K. Pradhan, "Fuzzy Linguistic Approach to Matchmaking in E-Commerce," *International Conference on Information Technology Proceedings,* 2005.

[18] A. Ragone, U. Straccia, T. Di Noia, E. Di Sciascio and F. M. Donini, "Vague Knowledge Bases for Matchmaking in P2P E-Marketplaces," *ESWC,* 2007.

[19] G. Salton, *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer.* Addison-Wesley, 1989,

[20] M. Ströbel and M. Stolze, "A Matchmaking Component for the Discovery of Agreement and Negotiation Spaces in Electronic Markets," *Group Decision and Negotiation,* vol. 11, pp. 165-181, 2002.

[21] V. S. Subrahmanian, P. Bonatti, J. Dix, T. Eiter, S. Kraus, F. Ozcan and R. Ross, "Heterogenous Agent Systems," *MIT Press,* 2000.

[22] K. Sycara, S. Widoff , M. Klusch and J. Lu, "Larks: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace," *Autonomous Agents and Multi-Agent Systems,* vol. 5, pp. 173-203, 2002.

[23] D. Veit, J. P. Müller and C. Weinhardt, "Multidimensional Matchmaking for Electronic Markets," *International Journal of Applied Artificial Intelligence,* vol. 16, pp. 853-869, 2002.

[24] Z. Zhang and C. Zhang, "An Improvement to Matchmaking Algorithms for Middle Agents," *Procceding of International Conference on Autonomous Agents,* pp. 1340-1347, 2002.

APPENDIX-1

Each of the profiles P-1 to P-6 is matched with profiles P-8 to P-14 to obtain similarity values. All these profiles are obtained from an online free local classifieds service available at ‘http://fredericton.kijiji.ca’.
Only those matching profiles are displayed in the result where similarity value is non-zero.

House Owner’s Profile		
P-1 <bedrooms,{4}, No, 1> <laundry,{yes}, No, 1> <lease,{1-year}, No, 1> <rent, {1700}, No, 1> <type,{apartment},No,1>	**P-2** <available,{Sept-1},No,1 <pets, {no}, No, 1> <rent, {395}, No,1> <smoke,{no}, No,1> <type,{bachelor},No, 1>	**P-3** <available,{Sept-1},No,1 <bedrooms,{3},No, 1> <rent, {600-900}, No,1> <security,{700},No, 1> <type,{apartment},No,1>
P-4 <bedrooms,{1},No,1> <parking,{1}, No,1> <rent, {625}, No, 1> <type,{apartment},No,1>	**P-5** <rent, {300},No, 1> <type, {room}, No, 0.99>	**P-6** <available,{Aug-1},No,1 <bedrooms,{2}, No, 1> <laundry,{yes}, No, 1> <parking,{2}, No, 1> <rent, {900}, No, 1> <type,{apartment},No,1>

Buyer – Tenant’s Profile		
P-8 <available,{Sept-1}, No, 1> <bedrooms,{1}, No, 1> <rent, {100-400}, No, 1>	**P-9** <available, {Sept-1}, No, 1> <rent, {375}, No, 1> <type,{room},No, 1>	**P-11** <bedrooms,{ 2}, No, 1> <kids,{yes}, No, 1> <pets, {yes}, No, 1>

<type,{bachelor, room}, No,1>		<rent, {500}, No, 1> <type,{apartment},Yes,1>
P-12 <available,{Sept-1},No, 1> <parking,{1}, Yes, 1> <rent,{500}, No, 1> <type,{bachelor}, No, 1>	**P-13** <pets, {yes}, No, 1> <rent, {0},Yes, 1> <type,{room}, Yes, 1>	**P-14** <area,{downtown},No, 1> <available, {Sept-1}, No,1> <bedrooms,{2}, No, 1> <kids, {no}, No, 1> <laundry,{yes},No, 1> <pets,{yes}, No, 1> <rent, {800}, Yes, 1> <type,{apartment},No,1>

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 4 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

APPENDIX-2

Following example elaborates how the matchmaking algorithm calculates similarity values. Profile P1 is matched with P8, P13, P14 and P11 respectively.

P-1: <bedrooms,{4}, No, 1> <laundry,{yes}, No, 1> <lease,{1-year}, No, 1> <rent, {1700}, No, 1> <type,{apartment},No,1>

P-1 Vs P8

P-8 : <available,{Sept-1}, No, 1> <bedrooms,{1}, No, 1> <rent, {100-400}, No, 1> <type,{bachelor, room}, No,1>

Here we have a mismatch of 3 Hard constraints. The attributes bedrooms, rent and type are hard constraints and description values of these two profiles mismatch. Hence the **Similarity Value: 0.0**

P-1 Vs P13

P-13 : <pets, {yes}, No, 1> <rent, {0},Yes, 1> <type,{room}, Yes, 1>

Some attributes like bedrooms, laundry, lease from P-1 are not present in P-13 and attribute pets from P-13 is missing in P-1. But rent and type are soft constraint in one profile (P-13). Hence we have two Compromised matches of soft constraints. Hence the **Similarity Value: 0.9412**

P-1 Vs P14

P-14 : <area,{downtown},No, 1> <available, {Sept-1}, No,1> <bedrooms,{2}, No, 1> <kids, {no}, No, 1> <laundry,{yes},No, 1> <pets,{yes}, No, 1> <rent, {800}, Yes, 1> <type,{apartment},No,1>

The attributes bedrooms and type are hard constraints and description values of these two profiles mismatch. It is a mismatch of 2 Hard constraints and there are two compromised matches (laundry and rent). **Similarity Value: 0.397635**

P-1 Vs P11

P-11: <bedrooms,{ 2}, No, 1> <kids,{yes}, No, 1> <pets, {yes}, No, 1> <rent, {500}, No, 1> <type,{apartment},Yes,1>

The attributes bedrooms and rent are hard constraints and description values of these two profiles mismatch. So in total there is a mismatch of 2 Hard constraints but only one compromised match (type). Hence similarity value of this match is less than P1 Vs P14. **Similarity Value:0.1396**

Table-2: Cross dimensional analysis of various KRMs used for matchmaking systems

KRM	Matchmaking System	Types of Constraints supported	Result Classification Categories	Algorithm Details Matching Process
Array	COIN (1996) [12]	None	• Uses a distance measure of IR. • Returns matching document vector, including the name of the document.	IR based process – No Ontologies used The SMART [19] information retrieval system is used, to process and match free text and document vectors. A local concept corpus is maintained but it does not implement notion of semantic matchmaking,
	GRAPPA (2001) [23]	Hard / Soft Alternate Value	• Identifies k nearest requests based on distance function	Parallel amenable, IR based matching Profiles represented in XML Do not support N:M matching
Knowledge Representation Languages	LARKS based RETSINA System (2002) [22]	Range Value Alternate Value	• Exact, plug-in, relaxed. • Generate ranked list based on similarity value.	Performs syntactic and semantic matching. Uses IR technique to compare profiles. Concepts are stored using Information Terminological Language (ITL). Different matching modes apply different combination of filters for matchmaking
	Description Logic based NeoClassic (2004) [5]	Range Value Preferential (Weights are used to increase relevance of concept) Alternate Value	• Exact, Potential, Partial • Ranking of matching profiles	Rankpartial and rankpotential algorithm produce a distance based rank when a demand is matched with several supplies. Ontologies are used
	Web Service Technology DAML-S (2004) [13]	Range Value Alternate Value	• Exact, plug-in, subsume, intersection, disjoint.	A Description Logic reasoner, RACER, is used to compute semantic matches. DAML-S service ontology is used for service description.
Database	Match making in Learning Networks (2007) [14]	None	• List of all resources matching to the requested service are populated. • Matching criteria are- sharing and content competence, availability etc.	Especially used for matching resources in Learning Network. Database stores details of learning contents, learner information and available resources. Latent Semantic Analysis (LSA) is used match the request with resources.

Table-2: Cross dimensional analysis of various KRMs used for matchmaking systems (Continue…)

KRM	Matchmaking System	Types of Constraints supported	Result Classification Categories	Algorithm Details Matching Process
Tree	A Weighted-Tree (2004) [3]	Range Value Preferential Alternate Value	• A list of matching profiles is displayed.	Information of demand and supply is stored in node-labeled, arc-labeled, arc-weighted tree. A tree matching algorithm computes similarity.
Graph	An Ontology Driven Matchmaking Process (2004) [15]	Range Value - Not Clear Preferential - Not clear Alternate Value - Not clear	• A path set is determined after matching resumes with advertisement. • Weights assigned to relationships and nodes, are used to calculate scores. • System returns k best resumes.	Skill ontology is maintained as a graph. Nodes represent skills (hard and soft) and edge represents inheritance relationship between nodes.
Hybrid Combination of Description Languages, Fuzzy Rules and Utility theory	Vague Knowledge Bases for Matchmaking in P2P E-Marketplaces (2007) [18]	Hard / Soft Range Value (Discrete values) Preferential (Use of utility theory) Alternate Value	• Returns top k matching profiles based on the score.	Use of Fuzzy predicates supports vague rules Semantic matchmaking
Proposed (set of Nodes)	Hunt ForTune matchmaking system (2009) [11]	Hard / Soft Range Value Preferential Alternate Value Hidden Cost	• Similarity value • Six categories	Scalable algorithm Amenable for Parallelization Parameter supported