

# A Non-Monotonic Approach to Semantic Matchmaking and Request Refinement in E-Marketplaces

Tommaso Di Noia<sup>1</sup>, Eugenio Di Sciascio<sup>1</sup>, Francesco M. Donini<sup>2</sup>

<sup>1</sup> Politecnico di Bari, Via Re David, 200, I-70125, Bari, Italy  
{t.dinoia, disciascio}@poliba.it

<sup>2</sup> Università della Tuscia, via San Carlo, 32, I-01100, Viterbo, Italy  
donini@unitus.it

**Abstract.** We present and motivate a formal approach and algorithms for semantic matchmaking in an e-commerce framework. The proposed solution exploits nonmonotonic inferences for explanation services in the query-retrieval-refinement loop, and defines some guidelines to compute semantic-based ranking of offers.

## 1 Introduction

Matchmaking is, roughly speaking, the process of finding a list of appealing resources with respect to a given request. The Semantic Web initiative has launched a new challenge to this field: semantic matchmaking. It is the process of finding appealing resource taking into account also the semantics of resources annotation and request definition.

Studies on matchmaking go a long way back<sup>3</sup>. Here we concentrate only on relevant work in semantic matchmaking. To the best of our knowledge, [11, 13] subsumption based matchmaking was introduced for the first time introducing the idea of match classes based on logic entailment and satisfiability. In [19] and [18] a language, LARKS, was proposed specifically designed for agent advertisement. The matching process was carried out through five progressive stages, going from classical IR analysis of text to semantic match via  $\Theta$ -subsumption. The notion, inspired by Software Engineering, of *plug-in* match class was introduced to overcome in some way the limitations of a matching approach based on exact match. No ranking was devised but for what is called relaxed match, which basically reverts again to a IR free-text similarity measure and a basic service of a semantic approach, such as inconsistency check, seems unavailable with this type of match. In [20] a matchmaking framework was proposed, which operated on service descriptions in DAML+OIL and was based on the FaCT reasoner. An extension to the approach in [18] was proposed in [15] where two new levels for service profiles matching are introduced. There, the *intersection*

---

<sup>3</sup> For a detailed report on general matchmaking issues and systems we refer the interested reader to [16].

*satisfiable* match class is introduced, whose definition corresponds to the one of *potential matching* proposed in [9]. The approach presented does not introduce a ranking method to measure proximity of service descriptions.

In [9, 10] properties that a matchmaker should have in a DL based framework, were described and motivated, and algorithms to classify and rank matches into classes were presented. In [4, 3] the Difference Operator in DLs was proposed for matchmaking in the framework of web services. The approach uses the Concept Difference, followed by a set covering operation optimized using hypergraph techniques.

The need to overcome simple deductive based semantic-matchmaking is now increasingly acknowledged. Recent approaches try to tackle this issue adopting fuzzy-DLs as in Smart [1] or hybrid approaches, as in the OWLS-MX matchmaker [14]. Such approaches, anyway, relaxing the logical constraints, do not allow any explanation or automated revision service.

The remaining of this paper is structured as follows: in the next section, with the aid of simple propositional logic, we illustrate and motivate the rationale of our approach to semantic matchmaking and query refinement; we move on to more expressive Description Logics (DLs) in Section 3 and present our theoretical approach and algorithms for matchmaking and query refinement in DLs. Last section draws the conclusions.

## 2 Logic-based Matchmaking principles

Matchmaking is a widely used term in a variety of frameworks, comprising several –quite different– approaches. We begin this section trying to provide a generic and sound definition of matchmaking.

**Definition 1 (Matchmaking).** *Matchmaking is an information retrieval task whereby queries (a.k.a. demands) and resources (a.k.a. supplies) are expressed using semi-structured text in the form of advertisements, and task results are ordered (ranked) lists of those resources best fulfilling the query.*

This simple definition implies that –differently from classical unstructured-text Information Retrieval systems– some structure in the advertisements is expected in a matchmaking system, and matchmaking does not consider a fixed Database-oriented structure. Furthermore, usually database systems provide answers to queries that do not include a relevance ranking, which should be instead considered in a matchmaking process.

**Definition 2 (Semantic Matchmaking).** *Semantic matchmaking is a match-making task whereby queries and resources advertisements are expressed with reference to a shared specification of a conceptualisation for the knowledge domain at hand, i.e., an ontology.*

From now on, we concentrate on matchmaking in marketplaces, adopting specific terminology, to ease presentation of the approach. Nevertheless our approach applies to generic matchmaking of semantically annotated resources.

## 2.1 Foundation of Logical Matchmaking

Our research is inspired by the application of Description Logics to E-Commerce. However, we believe that foundations of a logical approach to matchmaking are more evident in a simple propositional setting first, then we will move on to DLs.

As usual, an interpretation assigns values *true*, *false* to elements of a propositional ontology  $\mathcal{T}$ , and truth values are assigned to formulae according to usual truth tables for  $\wedge, \vee, \neg, \Rightarrow, \Leftrightarrow$ . A set of formulae is *consistent* if there is at least one interpretation assigning *true* to all formulae.

Satisfiability and classification w.r.t. an ontology  $\mathcal{T}$  can be formally explained in terms of interpretations for a logical theory. In fact if we reduce an ontology to its logical axioms, then we can formalize both classification and satisfiability in terms of logical interpretations of a theory  $\mathcal{T}$ . In the following we will use  $C \Rightarrow_{\mathcal{T}} D$  for  $\mathcal{T} \models C \Rightarrow D$ .

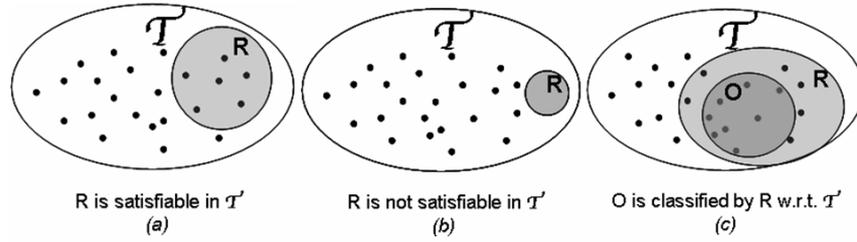


Fig. 1. Satisfiability and Classification

A request  $R$  (conversely a resource  $O$ ) is satisfiable w.r.t.  $\mathcal{T}$  if there is at least one interpretation in all the interpretations for  $\mathcal{T}$  which is also an interpretation for  $R$  (conversely for  $O$ ) – see Figure 1(a-b). For what concerns classification, given  $R$  and  $O$  both satisfiable w.r.t.  $\mathcal{T}$ , we say that  $O$  is classified by  $R$  if all the interpretations for  $O$  are also interpretations for  $R$  – see Figure 1(c).

Formally, let  $\mathcal{M}$  be the interpretations for  $\mathcal{T}$  and  $\mathcal{M}_R$  the interpretations in  $\mathcal{M}$  that satisfy the request  $R$  (respectively  $\mathcal{M}_O$  for the resource  $O$ ). We have  $R$  (conversely  $O$ ) is satisfiable if  $\mathcal{M}_R \neq \emptyset$  and  $R$  classifies  $O$  if  $\mathcal{M}_O \subseteq \mathcal{M}_R$ .

Given  $R$  and  $O$  both satisfiable w.r.t. an ontology, logic based approaches to matchmaking proposed in the literature [18, 15, 10] use classification and satisfiability to grade match results in five categories:

1. **Exact.** Figure 2(a).  $\mathcal{M}_R = \mathcal{M}_O$  – All the interpretations for  $R$  are also interpretations for  $O$  – in formulae  $\mathcal{T} \models R \Leftrightarrow O$ .
2. **Full - Subsumption.** Figure 2(b).  $\mathcal{M}_O \subseteq \mathcal{M}_R$  – All the interpretations for  $O$  are also interpretations for  $R$  – in formulae  $\mathcal{T} \models O \Rightarrow R$ .

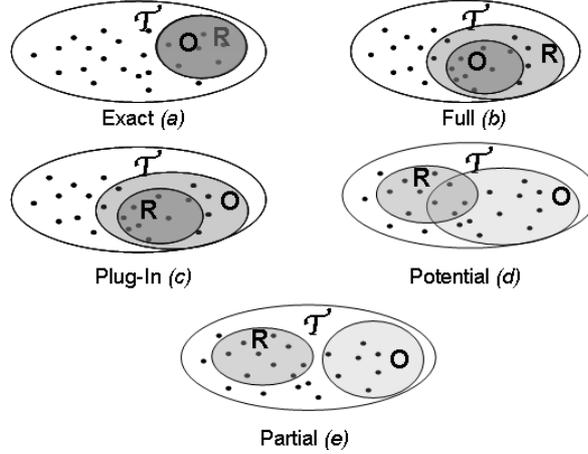


Fig. 2. Match Types

3. **Plug-In.** Figure 2(c).  $\mathcal{M}_R \subseteq \mathcal{M}_O$  – All the interpretations for R are also interpretations for O – in formulae  $\mathcal{T} \models R \Rightarrow O$ .
4. **Potential - Intersection.** Figure 2(d).  $\mathcal{M}_R \cap \mathcal{M}_O \neq \emptyset$  – Some interpretations for R are also interpretations for O – In formulae  $\mathcal{T} \not\models \neg(R \wedge O)$ .
5. **Partial - Disjoint.** Figure 2(e).  $\mathcal{M}_R \cap \mathcal{M}_O = \emptyset$  – No interpretations for R are also interpretations for O – In formulae  $\mathcal{T} \models \neg(R \wedge O)$ .

The best match type is obviously the first one – **Exact** – because both R and O express the same preferences; then we have **Full** match where all the interpretations for O are surely also interpretations for R and then O completely satisfies R; **Plug-In** match expresses the situation when O is more generic than R and then it is possible that the latter can be satisfied by the former; with **Potential** match we can only say that a similarity degree exists between O and R and then O may potentially satisfy R; finally Partial match states that there is no common interpretation between R and O. The following example shows, in a propositional logic setting, the rationale of such categorization.

Suppose to have the following set  $\mathcal{T}$  of propositional axioms modeling an automotive domain:

$$\mathcal{T} = \begin{cases} \text{ReflexSilver} \Leftrightarrow \text{Silver} \wedge \text{Metallic} \\ \text{PassengerAirbag} \Rightarrow \text{Airbag} \\ \text{EcoDiesel} \Rightarrow \text{Diesel} \\ \text{Gasoline} \Leftrightarrow \neg \text{Diesel} \\ \text{SatelliteAlarm} \Rightarrow \text{AlarmSystem} \end{cases}$$

Now imagine to have a request R and five offers  $O_{par}$ ,  $O_{pot}$ ,  $O_{full}$ ,  $O_{plug}$ ,  $O_{ex}$  as in the following:

$\mathbf{R} = \text{Sedan} \wedge \text{PassengerAirbag} \wedge \text{EcoDiesel} \wedge \text{ReflexSilver} \wedge \text{AlarmSystem}$   
 $\mathbf{O}_{par} = \text{Sedan} \wedge \text{Gasoline} \wedge \text{AlarmSystem} \wedge \text{PassengerAirbag} \wedge \text{ReflexSilver}$   
 $\mathbf{O}_{pot} = \text{Sedan} \wedge \text{Airbag} \wedge \text{Silver} \wedge \text{SatelliteAlarm} \wedge \text{EcoDiesel}$   
 $\mathbf{O}_{full} = \text{Sedan} \wedge \text{PassengerAirbag} \wedge \text{EcoDiesel} \wedge \text{ReflexSilver} \wedge \text{SatelliteAlarm}$   
 $\mathbf{O}_{plug} = \text{Sedan} \wedge \text{PassengerAirbag} \wedge \text{Diesel} \wedge \text{ReflexSilver} \wedge \text{AlarmSystem}$   
 $\mathbf{O}_{ex} = \text{Sedan} \wedge \text{PassengerAirbag} \wedge \text{EcoDiesel} \wedge \text{Metallic} \wedge \text{Silver} \wedge \text{AlarmSystem}$

In Table 1 are shown the interpretations of  $\mathcal{T}$  satisfying  $\mathbf{R}$ ,  $\mathbf{O}_{par}$ ,  $\mathbf{O}_{pot}$ ,  $\mathbf{O}_{full}$ ,  $\mathbf{O}_{plug}$  and  $\mathbf{O}_{ex}$ .

	Sedan	PassengerAirbag	EcoDiesel	ReflexSilver	Gasoline	Airbag	Silver	Metallic	Diesel	AlarmSystem	SatelliteAlarm
$\mathcal{M}_{\mathbf{R}}$											
T	T	T	T	T	F	T	T	T	T	T	T
T	T	T	T	T	F	T	T	T	T	T	F
$\mathcal{M}_{\mathbf{O}_{par}}$											
T	T	F	T	T	T	T	T	T	F	T	T
T	T	F	T	T	T	T	T	T	F	T	F
Partial Match: $\mathcal{M} \cap \mathcal{M}_{\mathbf{O}_{par}} = \emptyset$											
$\mathcal{M}_{\mathbf{O}_{pot}}$											
T	T	T	T	F	F	T	T	T	T	T	T
T	T	T	F	F	T	T	F	T	T	T	T
T	F	T	T	F	T	T	T	T	T	T	T
T	F	T	F	F	T	T	F	T	T	T	T
Potential Match: $\mathcal{M}_{\mathbf{R}} \cap \mathcal{M}_{\mathbf{O}_{pot}} \neq \emptyset$											
$\mathcal{M}_{\mathbf{O}_{full}}$											
T	T	T	T	F	T	T	T	T	T	T	T
Full Match: $\mathcal{M}_{\mathbf{O}_{full}} \subseteq \mathcal{M}_{\mathbf{R}}$											
$\mathcal{M}_{\mathbf{O}_{plug}}$											
T	T	T	T	F	T	T	T	T	T	T	T
T	T	T	T	F	T	T	T	T	T	T	F
T	T	F	T	F	T	T	T	T	T	T	T
T	T	F	T	F	T	T	T	T	T	T	F
Plug-In Match: $\mathcal{M}_{\mathbf{R}} \subseteq \mathcal{M}_{\mathbf{O}_{plug}}$											
$\mathcal{M}_{\mathbf{O}_{ex}}$											
T	T	T	T	F	T	T	T	T	T	T	T
T	T	T	T	F	T	T	T	T	T	T	F
Exact Match: $\mathcal{M}_{\mathbf{R}} = \mathcal{M}_{\mathbf{O}_{ex}}$											

**Table 1.** Matches

Actually, it is questionable whether a Plug-In match type should be considered better than Potential one. We note that some researchers also consider Plug-In match more favorable than full match (*e.g.*, see [15, 18]), motivating this choice with the idea that if  $\mathcal{M}_{\mathbf{R}} \subseteq \mathcal{M}_{\mathbf{O}}$  one may expect that the advertiser offering resource  $\mathbf{O}$  will probably have also more specific resources; in an e-commerce setting if the advertiser offers a *sedan* car it will also probably offer specific types of *sedans*. Nevertheless we argue that this idea prevents a fully automated matchmaking, which is possible when  $\mathcal{M}_{\mathbf{O}} \subseteq \mathcal{M}_{\mathbf{R}}$ , and furthermore it favors underspecified resource description, *i.e.*, an advertisement offering a sedan will always Plug-In match any request for a specific sedan, but will leave on the requester the burden to determine the right one – if any is actually available – for his/her needs. Even though Exact match is surely the best match, Full match might be considered –not always anyway– equivalent from the requester point

of view, because it states that at least all the features specified in  $R$  are also expressed in  $O$ .

Largest part of logic-based approaches only allow, as shown before, a categorization within match types. But while exact and full matches can be rare (and basically equivalent), a user may get several potential and partial matches. Then a useful logic-based matchmaker should provide a –logic– ordering of available resources vs. the request, but what we get using classification and satisfiability is a boolean answer. Also partial matches, as pointed out in [10], might be just “near miss”, *e.g.*, maybe just one requirement is in conflict, but a pure satisfiability check returns a hopeless *false* result, while it could be interesting to order “not so bad” offers according to their similarity to the request.

One may be tempted to revert to classical and well assessed Information Retrieval (IR) algorithms to get a rank for approximate matches (*e.g.*, so-called Hybrid approaches [14]), but regardless of well-known limits of unstructured text retrieval, there is something IR algorithms cannot do, while a logic approach can: provide explanations for matches result and suggest revision of requests.

## 2.2 Penalty Functions

Matches classification based on implication and satisfiability is still a coarse one, relying directly on known logical relations between formulae. In fact, the result of matchmaking should be a *rank* of resources/supplies/counteroffers, according to some criteria – possibly explicit – so that a user trusting the system would know whom to contact first, and in case of failure, whom next, and so on. Such a ranking process should satisfy some criteria that a Knowledge Representation approach suggests. We formulate ranking requirements by referring to properties of penalty functions  $p(O, R, \mathcal{T})$ ,  $O, R$  being two formulae and  $\mathcal{T}$  an ontology. We use penalty functions to rank  $O$  for a given  $R$  w.r.t. a ontology  $\mathcal{T}$ . Intuitively, for two given  $O_1, O_2$  in the marketplace, if  $p(O_1, R, \mathcal{T}) < p(O_2, R, \mathcal{T})$  then the issuer of demand  $R$  should rank  $O_1$  better than  $O_2$  when deciding whom to contact first. Clearly, a 0-penalty should be ranked best, and counteroffers with the same penalties should be ranked breaking ties.

The first property we recall is Non-symmetric evaluation of proposals.

**Definition 3.** A penalty function  $p(\cdot, \cdot, \cdot)$  is non-symmetric if there exists formulae  $R, O$  and an ontology  $\mathcal{T}$  such that  $p(O, R, \mathcal{T}) \neq p(R, O, \mathcal{T})$ .

This property is evident when all constraints of  $R$  are fulfilled by  $O$  but not vice versa. Hence,  $O$  should be among the top-ranked counteroffers in the list of potential partners of  $R$ , while  $R$  should *not* necessarily appear at the top in the list of potential partners of  $O$ .

Secondly, if logic is used to give some meaning to descriptions of supplies and demands, then proposals with the same meaning should be equally penalized, independently of their syntactic descriptions.

**Definition 4.** A penalty function  $p(\cdot, \cdot, \cdot)$  is syntax independent if for every triple of formulae  $O_1, O_2, R$ , and ontology  $\mathcal{T}$ , when  $\mathcal{T} \models O_1 \Leftrightarrow O_2$  then  $p(O_1, R, \mathcal{T}) =$

$p(O_2, R, \mathcal{T})$ , and the same holds also for the second argument, i.e.,  $p(R, O_1, \mathcal{T}) = p(R, O_2, \mathcal{T})$

Clearly, when the logic admits a normal form of expressions — as CNF or DNF for propositional logic, or the normal form of concepts for some DLs — using such a normal form in the computation of  $p(\cdot, \cdot, \cdot)$  ensures by itself syntax independence.

We now consider the relation between penalties and implication. We separately consider penalty functions for ranking potential matches —  $p_{\Rightarrow}(\cdot, \cdot, \cdot)$  — from those for ranking partial (conflicting) matches —  $p_{\emptyset}(\cdot, \cdot, \cdot)$ .

**Definition 5.** *A penalty function for potential matches is monotonic over implication whenever for every issued demand  $R$ , for every pair of counteroffers  $O_1$  and  $O_2$ , and ontology  $\mathcal{T}$ , if  $O_1$  and  $O_2$  are both potential matches for  $R$  w.r.t.  $\mathcal{T}$ , and  $(O_1 \Rightarrow_{\mathcal{T}} O_2)$ , then  $p_{\Rightarrow}(O_1, R, \mathcal{T}) \leq p_{\Rightarrow}(O_2, R, \mathcal{T})$*

Intuitively, the above definition could be read of as: if  $O_1 \Rightarrow_{\mathcal{T}} O_2$  then  $O_1$  should be penalized (and then ranked) either the same, or better than  $O_2$ . In a phrase, *A ranking of potential matches is monotonic over implication if the more specific, the better.* A dual property could be stated for the second argument: if  $R_1 \Rightarrow_{\mathcal{T}} R_2$  then a counteroffer  $O$  is less likely to fulfill all characteristics required by  $R_1$  than  $R_2$ . However, since our scenario is: “given an issuer of a demand  $R$  looking for a match in the marketplace, rank all possible supplies  $O_1, O_2, \dots$ , from the best one to the worst”, we do not deal here with this duality between first and second argument of  $p_{\Rightarrow}(\cdot, \cdot, \cdot)$ .

When turning to partial matches, in which some properties are already in conflict between supply and demand, the picture reverses. Now, adding another characteristic to an unsatisfactory proposal may only worsen this ranking (when another characteristic is violated) or keep it the same (when the new characteristic is not in conflict).

**Definition 6.** *A penalty function for partial matches is antimonotonic over implication whenever for every issued demand  $R$ , for every pair of supplies  $O_1$  and  $O_2$ , and ontology  $\mathcal{T}$ , if  $O_1$  and  $O_2$  are both partial matches for  $R$  w.r.t.  $\mathcal{T}$ , and  $(O_1 \Rightarrow_{\mathcal{T}} O_2)$ , then  $p_{\emptyset}(O_1, R, \mathcal{T}) \geq p_{\emptyset}(O_2, R, \mathcal{T})$*

Intuitively, if  $O_1 \Rightarrow_{\mathcal{T}} O_2$  then  $O_1$  should be penalized (and then ranked) either the same, or worse than  $O_2$ . In other words, *A ranking of partial matches is antimonotonic over implication if the more specific, the worse.* The same property should hold also for the second argument, since conjunction is commutative.

### 2.3 From Partial to (quasi-)Exact Match

We illustrate the rationale of our approach by computing what is needed in order to “climb the classification list” and reach a **Full** or an **Exact** match.

In particular, if we get a Partial match we could revise  $R$  relaxing some restrictions, in order to reach a Potential match. Once we get a Potential match

we can hypothesize what is not specified in  $O$  in order to reach a Full match and subsequently we can suggest to the user what is not specified in the relaxed  $R$  but it is in  $O$ . The ideal sequence should be then:

**Partial**  $\rightarrow$  **Potential**  $\rightarrow$  **Full** ( $\rightarrow$  **Exact**)

With reference to our previous example we show how this process can be performed in a propositional framework. Let us define the following  $R$  and  $O$  we will use in the rest of the Section:

**R** = Sedan  $\wedge$  PassengerAirbag  $\wedge$  EcoDiesel  $\wedge$  ReflexSilver  
**O** = Sedan  $\wedge$  Gasoline  $\wedge$  AlarmSystem  $\wedge$  Silver

It is easy to understand that due to the axiom Gasoline  $\Leftrightarrow \neg$ Diesel in  $\mathcal{T}$ ,  $\mathcal{M}_R \cap \mathcal{M}_O = \emptyset$  and then a Partial match occurs.

In this example we can easily identify the source of inconsistency and notice it is due to the EcoDiesel specification in  $R$  and the Gasoline one in  $O$ . So if the requester relaxes  $R$  giving up EcoDiesel specification, then we have a new contracted request. We call this new request  $K$  for **Keep**.

**K** = Sedan  $\wedge$  PassengerAirbag  $\wedge$  ReflexSilver

With respect to  $O$  now we have a Potential match. That is  $\mathcal{M}_K \cap \mathcal{M}_O \neq \emptyset$ .

Now to reach a Full match we should reduce  $\mathcal{M}_O$ . This is possible hypothesizing some unspecified characteristic  $H$  (for Hypothesis) in  $O$  such that  $\mathcal{M}_{O \wedge H} \subseteq \mathcal{M}_R$ . If we hypothesize  $H = \text{Metallic} \wedge \text{PassengerAirbag}$  the previous set relation holds. Then, having

$O \wedge H = \text{Sedan} \wedge \text{Gasoline} \wedge \text{AlarmSystem} \wedge \text{Silver} \wedge \text{Metallic} \wedge \text{PassengerAirbag}$

a Full match occurs *i.e.*,  $\mathcal{M}_{O \wedge H} \subseteq \mathcal{M}_K$ .

The previous example shows that some revision and hypotheses are needed in order to perform an extended matchmaking process and move through match classes.

**Partial**  $\rightarrow$  **Potential**. Contract  $R$  to  $K$  giving up elements  $G$  conflicting with  $O$  -Extend  $\mathcal{M}_R$  to  $\mathcal{M}_K$ .

Clearly, this is a non-monotonic task, particularly it can be reduced to contraction in a belief revision framework[12]. If you are in a Partial match with respect to and ontology  $\mathcal{T}$ , the following relation holds:  $\mathcal{T} \models \neg(R \wedge O)$ . Now consider  $\mathcal{T}'$  as an ontology equivalent to  $\mathcal{T}$  and such that  $\mathcal{T}' = \mathcal{T} \cup \{\rho \Leftrightarrow R, \omega \Leftrightarrow O\}$ , with  $\rho$  and  $\omega$  being two new propositional symbols. The previous relation can be rewritten as  $\mathcal{T}' \models \neg(\rho \wedge \omega)$ . In order to reach a Potential match you can give up some information in the axiom involving  $R$  and consider the contracted ontology  $\mathcal{T}'_c = \mathcal{T} \cup \{\rho \Leftrightarrow K, \omega \Leftrightarrow O\}$  such that  $\mathcal{T}'_c \not\models \neg(\rho \wedge \omega)$ .

**Potential**  $\rightarrow$  **Full**. Hypothesize missing characteristics  $H$  in  $O$  in order to completely satisfy  $K$ -Reduce  $\mathcal{M}_O$ .

Also this process can be modeled as a non-monotonic task. In particular, moving from a Potential match to a Full match, we use an abductive reasoning. In fact, we are not able to explain (satisfy in all the models)  $K$  given the manifestation  $O$  w.r.t. the ontology  $\mathcal{T}$ , and we look for a set of possible hypotheses  $H$  such that (1)  $\mathcal{T} \cup H \cup \{O\} \not\models \text{false}$  and (2)  $\mathcal{T} \cup H \cup \{O\} \models K$ , where the set of all possible hypotheses is, due to (1), a subset of the propositional alphabet used to model  $\mathcal{T}$ ,  $R$  and  $O$ .

While performing contraction and abduction, minimality criteria have to be taken into account. If we are a demander:

- we wish to keep as much as possible of the original request. Trivially if we contract all the request, we reach a Potential match with the supply, but it should be obvious that this is not a good solution from the demander point of view.
- we wish to hypothesize as less as possible features in the supply in order to be satisfied. We could hypothesize all the characteristics we requested in the supply, and reach a Full match. Again this is a trivial solution.

Based on the previous observation, if we have more than one supply during the contraction phase the demander should be inclined to choose the one with the minimum number of features to be contracted. During the abduction process the demander will move in favor of supplies with less requested features to be hypothesized in  $O$ .

Observe that we are not asking the requester to actually go through the whole process. Yet our approach has a twofold advantage: the requester can use provided information to actually revise her request but the information we extract is also all what is needed to compute a semantic distance between  $R$  and  $O$  to be used by a broker in an e-marketplace.

Once we know what has to be contracted and hypothesized it is possible to compute a match degree based on  $K$  and  $H$ , that is what is needed in order to reach a Full match. In case of multiple resources, we can use this match degree as a score to rank such resources with respect to  $R$ . Such a scoring function should be in the form:

$$match\_degree = \rho(R, O, K, H, \mathcal{T}) \quad (1)$$

$\rho$  combines all the causes for a non-Full match. Notice that  $\rho$  needs also  $\mathcal{T}$  to compute *match\_degree*; in fact in  $\mathcal{T}$  the semantics of  $K$ ,  $H$ ,  $R$  and  $O$  are modeled, which should be taken into account when evaluating how to weight them with respect to  $O$  and  $R$ .

Before making the final step beyond, moving from a Full to an Exact match, some considerations are needed.

In an *Open World* semantics, what is not specified in a formula has not to be interpreted as a constraint of absence. It is a "don't care" specifications. In a resource retrieval scenario – as a marketplace is – this can be due to basically two reasons:

- the user really does not care about the unspecified information.
- the user does not own that knowledge. She is not aware that it is possible to specify some other characteristics in the request, or she simply did not consider further possible specifications. She is not necessarily an expert of the marketplace knowledge domain.

In the second case a further refinement makes sense. A way to do this is to present to the user all the knowledge modeled in  $\mathcal{T}$  and ask her to refine the query, adding characteristics found in  $\mathcal{T}$ . This approach has at least two main drawbacks:

1. The user must be bored browsing all  $\mathcal{T}$  in order to find something interesting to be added to the request.
2. She can choose something in  $\mathcal{T}$  which is not in any offer in the marketplace. Then after the query refinement she would not see any change in the list ranked using  $\rho(\mathbf{R}, \mathbf{O}, \mathbf{K}, \mathbf{H}, \mathcal{T})$ .

To avoid the above drawbacks, we might suggest to the requester only those characteristics able to change the ranked list of offers within the marketplace. Then (in an ideal marketplace where the only offer is  $\mathbf{O}$ ) we could suggest to the user to refine the contracted request adding  $\mathbf{B}' = \mathbf{AlarmSystem} \wedge \mathbf{Gasoline}$ , where  $\mathbf{B}'$  (for Bonus) represents what is specified in  $\mathbf{O}$  but is not in  $\mathbf{K}$ . But notice that in  $\mathbf{B}'$  we have  $\mathbf{Gasoline}$ , that is the source of inconsistency of the original request  $\mathbf{R}$  with the original  $\mathbf{O}$ . Then it would be very strange if the user refined her request by adding something which is in conflict with her initial preferences. The user is likely to refine adding at most  $\mathbf{B} = \mathbf{AlarmSystem}$ .

**Full**  $\rightarrow$  **quasi-Exact**. Suggest to the requester what should be added to  $\mathbf{K}$  looking at non requested features  $\mathbf{B}$  (for bonus) in  $\mathbf{O}$  –Reduce  $\mathcal{M}_{\mathbf{K}}$ .

In order to propose only appealing features to the user, first of all it is necessary to remove from  $\mathbf{O}$  the causes for inconsistency with  $\mathbf{K}$ . This is clearly the contraction problem seen in action to move from Partial to Potential match with  $\mathbf{K}$  and  $\mathbf{O}$  flipped over. This time we want to contract  $\mathbf{O}$  w.r.t.  $\mathbf{K}$ . Once we have deleted the inconsistencies from  $\mathbf{O}$  reducing it to  $\mathbf{K}_{\mathbf{O}}$ , we need to make some hypotheses on how the user could refine the query so that the (contracted) request completely overlap the contracted supply  $\mathbf{K}_{\mathbf{O}}$ . Again an abduction problem has to be solved in order cope with this problem. We have to hypothesize  $\mathbf{B}$  such that  $\mathbf{K} \wedge \mathbf{B} \Rightarrow_{\mathcal{T}} \mathbf{K}_{\mathbf{O}}$ .

## 2.4 Penalty Functions and Non-Monotonic Tasks for Resources Ranking

In this section we will discuss relations between penalty functions introduced in Section 2.2 and the process of moving through match classes outlined in the previous Section.

A possible  $p_{\emptyset}(\cdot, \cdot, \cdot)$ , can measure how difficult it is to move from Partial to Potential match, or in other words how big is  $\mathbf{G}$  with respect to the original  $\mathbf{R}$ .

Looking at the contraction process highlighted in case of Partial match, it is easy to show that it is a non-symmetric process. If we contract R w.r.t. O, we obtain a new request K which is, generally, different from the contracted version of O w.r.t. R. Then property 3 is satisfied. Surely, a contraction reasoning task is syntax independent and then also property 4 is satisfied. For what concerns property 6, intuitively, if we add a new feature to R (or equivalently O) and it is not in conflict with O (or R), the information to give up G remain the same; if the new feature we add to R (O) is in conflict with O (R), then we have to give up this new feature and increase G.

For  $p_{\Rightarrow}(\cdot, \cdot, \cdot)$ , we might define a function computing how far is a Potential match from a Full match, *i.e.*, a function measuring H. The abduction process to move from Potential to Full match is both non-symmetric w.r.t. R and O and syntax independent, so properties 5 and 6 are satisfied. The last property to be satisfied is the monotonicity of  $p_{\Rightarrow}(\cdot, \cdot, \cdot)$ . Again, thinking at the abduction process we have that if we add in R (respectively O) a feature which is not implied by O (respectively in R), then H increases. If we add a feature which is implied by O (respectively by R), the hypotheses to be formulated are exactly the same and also H remains unchanged.

Based on the above consideration we can redefine  $\rho$ , computing an overall match degree of R w.r.t. O, as a function of  $p_{\emptyset}(\cdot, \cdot, \cdot)$  and  $p_{\Rightarrow}(\cdot, \cdot, \cdot)$ .

$$match\_degree = \rho(R, O, K, H, \mathcal{T}) = \rho(p_{\emptyset}(O, R, \mathcal{T}), p_{\Rightarrow}(O, R, \mathcal{T})) \quad (2)$$

Actually, if R and O are in a potential match, only  $p_{\Rightarrow}(O, R, \mathcal{T})$  contributes in computing the match degree. In this case  $\rho(p_{\emptyset}(O, R, \mathcal{T}), p_{\Rightarrow}(O, R, \mathcal{T})) = p_{\Rightarrow}(O, R, \mathcal{T})$ . Notice that modeling the matchmaking process as a sequence of non-monotonic reasoning tasks, and computing the match degree based on them, we are also able to provide explanations for the match degree. That is, when the system presents a ranked list of possible supplies to the requester, it is also able to justify the results – and then increase the trust level of the user in the system – but also guide the user in the refinement process.

### 3 DL inference services for matchmaking

Although useful to model simplified examples, propositional logic is obviously limited for what concerns expressiveness. In the following we will refer to Description Logics (DL) whose formal semantics is the basis of the Ontology Web Language OWL-DL [17], and model a DL-based framework to cope with the issues introduced here. Please refer to [2] for a comprehensive survey on DLs.

DL-based systems usually provide at least two basic reasoning services:

1. *Concept Satisfiability*:  $\mathcal{T} \models R \not\sqsubseteq \perp$  – Given a TBox  $\mathcal{T}$  and a concept R, does there exist at least one model of  $\mathcal{T}$  assigning a non-empty extension to R?
2. *Subsumption*:  $\mathcal{T} \models R \sqsubseteq O$  – Given a TBox  $\mathcal{T}$  and two concepts R and O, is R more general than O in any model of  $\mathcal{T}$ ?

Matchmaking services outlined in the previous section call for other, non-monotonic inference services, we briefly recall hereafter.

Let us consider concepts  $O$  and  $R$  and an ontology  $\mathcal{T}$ . If a partial match occurs, *i.e.*, they are not compatible with each other w.r.t.  $\mathcal{T}$ , one may want to retract specifications in  $R$ ,  $G$  (for *Give up*), to obtain a concept  $K$  (for *Keep*) such that  $K \sqcap O$  is satisfiable in  $\mathcal{T}$ . In [5] the Concept Contraction problem was defined as follows:

**Definition 7.** *Let  $\mathcal{L}$  be a DL,  $O, R$ , be two concepts in  $\mathcal{L}$  and  $\mathcal{T}$  be a set of axioms in  $\mathcal{L}$ , where both  $O$  and  $R$  are satisfiable in  $\mathcal{T}$ . A Concept Contraction Problem (CCP), identified by  $\langle \mathcal{L}, O, R, \mathcal{T} \rangle$ , is finding a pair of concepts  $\langle G, K \rangle \in \mathcal{L} \times \mathcal{L}$  such that  $\mathcal{T} \models R \equiv G \sqcap K$ , and  $K \sqcap O$  is satisfiable in  $\mathcal{T}$ . Then  $K$  is a contraction of  $R$  according to  $O$  and  $\mathcal{T}$ .*

If nothing can be kept in  $R$  during the contraction process, we get the worst solution — from a matchmaking point of view —  $\langle G, K \rangle = \langle R, \top \rangle$ , that is give up everything of  $R$ . If  $R \sqcap O$  is satisfiable in  $\mathcal{T}$ , that is a potential match occurs, nothing has to be given up and the solution is  $\langle \top, R \rangle$ , that is, give up nothing. Hence, a Concept Contraction problem amounts to an extension of a satisfiable one. Since usually one wants to give up as few things as possible, some minimality criteria in the contraction must be defined [12]. In most cases a pure logic-based approach could be not sufficient to decide between which beliefs to give up and which to keep. There is the need to model and define some extra-logical information, which have to be taken into account. For instance, one could be interested in contracting only some specification in her request, while others have to be considered strict [7].

If the offered resource  $O$  and the request  $R$  are in a potential match, it is necessary to assess what should be hypothesized  $H$  in  $O$  in order to completely satisfy  $R$  and then move to a full match. In [8] the Concept Abduction problem was defined as follows:

**Definition 8.** *Let  $\mathcal{L}$  be a DL,  $O, R$ , be two concepts in  $\mathcal{L}$ , and  $\mathcal{T}$  be a set of axioms in  $\mathcal{L}$ , where both  $O$  and  $R$  are satisfiable in  $\mathcal{T}$ . A Concept Abduction Problem (CAP), identified by  $\langle \mathcal{L}, R, O, \mathcal{T} \rangle$ , is finding a concept  $H \in \mathcal{L}$  such that  $\mathcal{T} \models O \sqcap H \sqsubseteq R$ , and moreover  $O \sqcap H$  is satisfiable in  $\mathcal{T}$ . We call  $H$  a hypothesis about  $O$  according to  $R$  and  $\mathcal{T}$ .*

Obviously the definition refers to satisfiable  $O$  and  $R$ , since  $R$  unsatisfiable implies that the CAP has no solution at all, while  $O$  unsatisfiable leads to counterintuitive results ( $\neg R$  would be a solution in that case). If  $O \sqsubseteq R$  then we have  $H = \top$  as a solution to the related CAP. Hence, Concept Abduction amounts to extending subsumption. On the other hand, if  $O \equiv \top$  then  $H \sqsubseteq R$ .

Concept Abduction and Concept Contraction can be used for respectively subsumption and satisfiability explanation. For Concept Contraction, having two concepts whose conjunction is unsatisfiable, in the solution  $\langle G, K \rangle$  to the CCP  $\langle \mathcal{L}, R, O, \mathcal{T} \rangle$ ,  $G$  represents "why"  $R \sqcap O$  are not compatible. For Concept Abduction, having  $R$  and  $O$  such that  $O \not\sqsubseteq R$ , the solution  $H$  to the CAP

$\langle \mathcal{L}, R, O, \mathcal{T} \rangle$  represents "why" the subsumption relation does not hold.  $H$  can be interpreted as *what is specified in R and not in O*.

## 4 Making the match

With respect to the match classification presented in Section 2 we show how it is possible to exploit both standard and non-standard inference services for DL in order to identify match classes and go from a Partial match to a Full (or Exact) match, and use the obtained information to provide a semantic-based score measuring similarity w.r.t. the request.

Using Subsumption and Concept Satisfiability we can rewrite match classes in terms of Description Logics. Given an ontology  $\mathcal{T}$  and a request and an offer, expressed as DL complex concepts,  $R$  and  $O$ , both satisfiable w.r.t.  $\mathcal{T}$ , we have:

$$\begin{aligned}
 \mathbf{Exact} : \mathcal{T} &\models R \equiv O \\
 \mathbf{Full} : \mathcal{T} &\models O \sqsubseteq R \\
 \mathbf{Plug-In} : \mathcal{T} &\models R \sqsubseteq O \\
 \mathbf{Potential} : \mathcal{T} &\not\models R \sqcap O \sqsubseteq \perp \\
 \mathbf{Partial} : \mathcal{T} &\models R \sqcap O \sqsubseteq \perp
 \end{aligned}$$

Both Concept Abduction and Concept Contraction can be used to suggest guidelines on what, given  $O$ , has to be revised and/or hypothesized to obtain a Full match with  $R$ .

### 4.1 Explanation for Ranking

**Partial**→**Potential** If  $R \sqcap O \equiv \perp$  – Partial match – then solving the related Concept Contraction Problem we have  $R \equiv G_R \sqcap K_R$  such that  $K_R \sqcap O \not\equiv \perp$  w.r.t.  $\mathcal{T}$ . That is, we contract  $R$  to  $K_R$  such that there is a Potential match between the contracted request and  $O$ .

**Potential**→**Full** Once we are in a Potential match, we can formulate hypotheses on what should be hypothesized in  $O$  in order to completely satisfy the contracted  $R$ . If we solve the related Concept Abduction Problem, we can compute an hypothesis  $H$  such that  $O \sqcap H \sqsubseteq K_R$  and reach a Full match with the contracted request.

The above concepts can be formalized in the following simple algorithm:

**Algorithm** *retrieve*( $R, O, \mathcal{T}$ )

**input**  $O, R \equiv K \sqcap G$  concepts satisfiable w.r.t.  $\mathcal{T}$

**output**  $\langle G, H \rangle$ , *i.e.*, the part in  $R$  that should be retracted and the part in  $O$  that should be hypothesized to have a full match between  $O$  and  $K$  (the contracted  $R$ )

**begin algorithm**

1: **if**  $\mathcal{T} \models R \sqcap O \sqsubseteq \perp$  **then**

```

2:    $\langle G, K \rangle = \text{contract}(O, R, \mathcal{T});$ 
3:    $H_K = \text{abduce}(O, K, \mathcal{T});$ 
4:   return  $\langle G, H \rangle;$ 
5: else
6:    $H = \text{abduce}(O, R, \mathcal{T});$ 
7:   return  $\langle \top, H \rangle;$ 
end algorithm

```

Notice that  $H = \text{abduce}(O, R, \mathcal{T})$  [rows 3,6] determines a concept  $H$  such that  $O \sqcap H \sqsubseteq R$ ,  $\langle G, K \rangle = \text{contract}(O, R, \mathcal{T})$  [row 2] determines two concepts  $G$  and  $K$  such that  $R \equiv G \sqcap K$  and  $\mathcal{T} \models K \sqcap O \not\sqsubseteq \perp$  following minimality criteria as suggested in [8, 5, 6]. Also notice that, given a CAP or a CCP, usually there is not only one single solution. But in order to provide a match explanation, in rows 2,3 and 6 only one solution has to be returned. In this case context-aware information may be used as criteria for solution selection.

## 4.2 Query Refinement

**Full**→**quasi-Exact** During this step, in order to overcome the suggestion of "fake bonuses", we have to identify which part of  $O$  generated the inconsistency with  $R$  before contracting. We can solve a Concept Contraction Problem between  $O$  and  $R$  contracting  $O$ . That is we have  $O \equiv G_O \sqcap K_O$  such that  $K_O \sqcap R \not\sqsubseteq \perp$  w.r.t.  $\mathcal{T}$ . In [8], among others, the conjunction minimal solution to a CAP is proposed for DLs admitting a normal form with conjunctions of concepts. A solution belonging to such solution is in the form  $B = \sqcap_{j=1..k} C_j$ , where  $C_j$  are DL concepts and is **irreducible**, *i.e.*,  $B$  is such that for each  $h \in 1, \dots, k$ ,  $\sqcap_{j=1..h-1, h+1..k} C_j$  is not a solution for the CAP.

In the following, the algorithm *computeBonus*( $O, R, \mathcal{T}$ ) is presented, able to compute what should be hypothesized in the requester preferences in order to get a better match result, and –if possible– an Exact match (see 2). It takes as input an offer  $O$ , a request  $R$  and the ontology  $\mathcal{T}$  they refer to.

**Algorithm** *computeBonus*( $O, R, \mathcal{T}$ )  
**input**  $O$  and  $R$  DL concepts both satisfiable w.r.t.  $\mathcal{T}$   
reference ontology  
**output**  $B_{irr}$  a set of DL concepts representing bonuses  
**begin algorithm**  
1:  $B = \emptyset;$   
2:  $B_{irr} = \emptyset;$   
3:  $\langle G_R, K_R \rangle = \text{contract}(O, R, \mathcal{T});$   
4:  $\langle G_O, K_O \rangle = \text{contract}(R, O, \mathcal{T});$   
5:  $B = \text{abduce}(K_R, K_O, \mathcal{T});$   
6: **for each**  $C_j \in B$   
7:  $B_{irr} = B_{irr} \cup \{C_j\};$   
8: **return**  $B_{irr};$   
**end algorithm**

The problem of *fake bonuses* is taken into account in rows 3-5 of *computeBonus*. In row 3, a Concept Contraction Problem is solved, contracting R in  $K_R$  and identifying in  $G_R$  the source of inconsistency with O. In row 4 the same is performed for O identifying in  $K_O$  the part of the offer which is compatible with R and in  $G_O$  the incompatible one and then likely to contain the *fake bonuses*. In row 5 we compute B, solution of Concept Abduction Problem such that  $K_R \sqcap B \sqsubseteq K_O$ . Notice that adding B to  $K_R$  we are neither in a Plug-In match nor in an Exact one with respect to the contracted request  $K_R$ . In fact, we would have a Plug-In match if  $K_R \sqcap B \sqsubseteq O$  rather than  $K_O$  and we could have an Exact match adding also *fake bonuses* which are isolated now in  $G_O$ .

## 5 Conclusion

In this paper we have motivated the need for nonmonotonic inference services, particularly abduction and contraction in a belief revision framework, for logic-based matchmaking and query refinement in e-commerce scenarios. We have also outlined some guidelines on how to exploit these services for matches explanation and ranking.

## References

1. S. Agarwal and S. Lamparter. smart - a semantic matchmaking portal for electronic markets. In *Proceedings of the 7th International IEEE Conference on E-Commerce Technology 2005*, 2005.
2. F. Baader, D. Calvanese, D. Mc Guinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2002.
3. B. Benatallah, M-S. Hacid, C. Rey, and F. Toumani. Request Rewriting-Based Web Service Discovery. In *International Semantic Web Conference*, volume 2870 of *Lecture Notes in Computer Science*, pages 242–257. Springer, 2003.
4. B. Benatallah, M-S. Hacid, C. Rey, and F. Toumani. Semantic Reasoning for Web Services Discovery. In *Proc. of Workshop on E-Services and the Semantic Web at WWW 2003*, May 2003.
5. S. Colucci, T. Di Noia, E. Di Sciascio, F.M. Donini, and M. Mongiello. Concept Abduction and Contraction in Description Logics. In *Proceedings of the 16th International Workshop on Description Logics (DL'03)*, volume 81 of *CEUR Workshop Proceedings*, September 2003.
6. S. Colucci, T. Di Noia, E. Di Sciascio, F.M. Donini, and M. Mongiello. Uniform Tableaux-Based Approach to Concept Abduction and Contraction in ALN DL. In *Proceedings of the 17th International Workshop on Description Logics (DL'04)*, volume 104 of *CEUR Workshop Proceedings*, 2004.
7. S. Colucci, T. Di Noia, E. Di Sciascio, F.M. Donini, and M. Mongiello. Concept Abduction and Contraction for Semantic-based Discovery of Matches and Negotiation Spaces in an E-Marketplace. *Electronic Commerce Research and Applications*, 4(4):345–361, 2005.
8. T. Di Noia, E. Di Sciascio, F.M. Donini, and M. Mongiello. Abductive matchmaking using description logics. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI 2003)*, pages 337–342, Acapulco, Mexico, August 9–15 2003. Morgan Kaufmann, Los Altos.

9. T. Di Noia, E. Di Sciascio, F.M. Donini, and M. Mongiello. Semantic matchmaking in a P-2-P electronic marketplace. In *Proc. Symposium on Applied Computing (SAC '03)*, pages 582–586. ACM, 2003.
10. T. Di Noia, E. Di Sciascio, F.M. Donini, and M. Mongiello. A system for principled Matchmaking in an electronic marketplace. In *Proc. International World Wide Web Conference (WWW '03)*, pages 321–330, Budapest, Hungary, May 20–24 2003. ACM, New York.
11. E. Di Sciascio, F.M. Donini, M. Mongiello, and G. Piscitelli. A Knowledge-Based System for Person-to-Person E-Commerce. In *Proceedings of the KI-2001 Workshop on Applications of Description Logics (ADL-2001)*, volume 44 of *CEUR Workshop Proceedings*, 2001.
12. P. Gärdenfors. *Knowledge in Flux: Modeling the Dynamics of Epistemic States*. Bradford Books, MIT Press, Cambridge, MA, 1988.
13. J. Gonzales-Castillo, D. Trastour, and C. Bartolini. Description Logics for Matchmaking of Services. In *Proceedings of the KI-2001 Workshop on Applications of Description Logics (ADL-2001)*, volume 44. CEUR Workshop Proceedings, 2001.
14. M. Klusch, B. Fries, M. Khalid, and Katia Sycara. Owls-mx: Hybrid owl-s service matchmaking. In *Proceedings of 1st Intl. AAAI Fall Symposium on Agents and the Semantic Web*, 2005.
15. L. Li and I. Horrocks. A Software Framework for Matchmaking Based on Semantic Web Technology. In *Proc. International World Wide Web Conference (WWW '03)*, pages 331–339, Budapest, Hungary, May 20–24 2003. ACM, New York.
16. 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*, 8(4):9–37, 2004.
17. OWL. Web Ontology Language. [www.w3.org/TR/owl-features/](http://www.w3.org/TR/owl-features/), 2004.
18. M. Paolucci, T. Kawamura, T.R. Payne, and K. Sycara. Semantic Matching of Web Services Capabilities. In *The Semantic Web - ISWC 2002*, number 2342 in *Lecture Notes in Computer Science*, pages 333–347. Springer-Verlag, 2002.
19. K. Sycara, S. Widoff, M. Klusch, and J. Lu. LARKS: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace. *Autonomous agents and multi-agent systems*, 5:173–203, 2002.
20. D. Trastour, C. Bartolini, and C. Priest. Semantic Web Support for the Business-to-Business E-Commerce Lifecycle. In *Proc. International World Wide Web Conference (WWW) '02*, pages 89–98. ACM, 2002.