



## BIROn - Birkbeck Institutional Research Online

Calì, Andrea and Calvanese, D. and Colucci, S. and Di Noia, T. and Donini, F.M. (2004) A description logic based approach for matching user profiles. In: Haarslev, V. and Moller, R. (eds.) Proceedings of the 2004 International Workshop on Description Logics. CEUR Workshop Proceedings.

Downloaded from: <http://eprints.bbk.ac.uk/id/eprint/42853/>

*Usage Guidelines:*

Please refer to usage guidelines at <https://eprints.bbk.ac.uk/policies.html> or alternatively contact [lib-eprints@bbk.ac.uk](mailto:lib-eprints@bbk.ac.uk).

# A Description Logic Based Approach for Matching User Profiles

Andrea Cali<sup>1,2</sup>, Diego Calvanese<sup>2</sup>, Simona Colucci<sup>3</sup>,  
Tommaso Di Noia<sup>3</sup> Francesco M. Donini<sup>4</sup>

<sup>1</sup>Dip. di Informatica e Sistemistica  
Università di Roma “La Sapienza”  
Via Salaria 113  
I-00198 Roma, Italy  
ac@andreacali.com

<sup>2</sup>Faculty of Computer Science  
Free University of Bolzano/Bozen  
Piazza Domenicani, 3  
I-39100 Bolzano, Italy  
calvanese@inf.unibz.it

<sup>3</sup>Dip. di Elettrotecnica ed Elettronica  
Politecnico di Bari  
Via Re David 200  
I-70125 Bari, Italy  
{s.colucci,t.dinoia}@poliba.it

<sup>4</sup>Università della Tuscia  
Facoltà di Scienze Politiche  
Via San Carlo 32,  
I-01100 Viterbo, Italy  
donini@unitus.it

## Abstract

Several applications require the matching of user profiles, e.g., job recruitment or dating systems. In this paper we present a logical framework for specifying user profiles that allows profile description to be incomplete in the parts that are unavailable or are considered irrelevant by the user. We present an algorithm for matching demands and supplies of profiles, taking into account incompleteness of profiles and incompatibility between demand and supply. We specialize our framework to dating services; however, the same techniques can be directly applied to several other contexts.

## 1 Introduction

The problem of matching demands and supplies of personal profiles arises in the business of recruitment agencies, in firms’ internal job assignments, and in the recently emerging dating services. In all scenarios, a list of descriptions of persons is to be matched with a list of descriptions of required persons. In electronic commerce, the general problem is known as *matchmaking*, although here we do not consider any exchange of goods or services.

We stress the fact that in matchmaking, finding an exact match of profiles is not the objective; in fact, such a match is very unlikely to be found, and in all cases where an exact match does not exist, a solution to matchmaking must provide one or more *best possible* matches to be explored. Non-exact matches should consider both missing information — details that could be positively assessed in a second phase — and conflicting information — details that should be negotiated if the proposed match

is worth enough pursuing. Moreover, when several matches are possible, a matchmaker should list them in a most-promising order, so as to maximize the probability of a successful match within the first trials. However, such an order should be based on transparent criteria — possibly, logic — in order for the user to trust the system.

Profiles matchmaking can be addressed by a variety of techniques, ranging from simple bipartite graph matching (with or without cost minimization) [9], to vector-based techniques taken from classical Information Retrieval [11, 13, 12], to record matching in databases, among others. We now discuss some drawbacks of these techniques when transferred to solve matchmaking.

Algorithms for bipartite graph matching find optimal solutions when trying to maximize the number of matches [8, 10]. However, such algorithms rely on some way of assigning *costs* to every match between profiles. When costs are assigned manually, knowledge about them is completely implicit (and subjective), and difficult to revise. Moreover, in maximizing the number of matches a system may provide a *bad* service to single end users: for example, person  $P_1$  could have a best match with job profile  $J_1$ , but she might be suggested to take job  $J_2$  just because  $J_1$  is the only available job for person  $P_2$ . Hence, from end user’s viewpoint, maximizing the number of matches is not the feature that a matchmaker should have.

Both Database techniques for record matching (even with null values), and information retrieval techniques using similarity between weighted vectors of stemmed terms, are not suited for dealing with incomplete information usually present in matchmaking scenarios. In fact, information about profiles is almost always incomplete, not only because some information is unavailable, but also because some details are simply considered irrelevant by either the supplier or the demander — and should be left as such. Imposing a system interface for entering profiles with long and tedious forms to be filled in, is the most often adopted “solution” to this incompleteness problem — but we consider this more an escape for constraining real data into an available technique, than a real solution. For example, in a job posting/finding system, the nationality could be considered irrelevant for some profiles (and relevant for others); or in a dating service, some people may find disturbing (or simply inappropriate) the request to specify the kind of preferred music, etc. In such situations, missing information can be assumed as an “any-would-fit” assertion, and the system should cope with this incompleteness as is.

To sum up, we believe that there is a *representation problem* that undermines present solutions to matchmaking: considering how profiles information is represented is a fundamental step to reach an effective solution, and representations that are either too implicit, or overspecified, lead to unsatisfactory solutions.

Therefore, our research starts with proposing a language, borrowed from Artificial Intelligence, that allows for incomplete descriptions of profiles, and both positive and negative information about profiles. In particular, we propose a Description Logic [1] specifically tailored for describing profiles. Then, we model the matching process as a special reasoning service about profiles, along the lines of [5, 6]. Specifically, we consider separately conflicting details and missing details, and evaluate how likely is the match to succeed, given both missing and conflicting details. Our approach makes transparent the way matches are evaluated — allowing end users to request

justifications for suggested matches. We devise some special-purpose algorithms to solve the problem for the language we propose, and evaluate the possible application scenarios of a dating service.

The paper is organized as follows. In Section 2 we present the Description Logic we use for describing profiles. In Section 3 we describe how to represent user profiles, and in Section 4 we present the algorithm for matching user profiles. Section 5 concludes the paper.

## 2 A Description Logic for Representing Profiles

We use a restriction of the  $\mathcal{ALC}(\mathcal{D})$  Description Logic, that, besides concepts and roles to represent properties of (abstract) objects, also allows one to express quantitative properties of objects, such as weight, length, etc., by means of *concrete domains* [2]. Each concrete domain  $\mathcal{D}$ , e.g., the real numbers  $\mathbb{R}$ , has a set of associated predicate names, where each predicate name  $p$  denotes a predicate  $p^{\mathcal{D}}$  over  $\mathcal{D}$ . For our purpose, it is sufficient to restrict the attention to unary predicates, and we assume that among such unary predicates we always have a predicate  $\top$  denoting the entire domain, and predicates  $\geq_{\ell}(\cdot)$  and  $\leq_{\ell}(\cdot)$ , for arbitrary values  $\ell$  of  $\mathcal{D}$ . We also assume that the concrete domains we deal with are *admissible*, which is a quite natural assumption, satisfied e.g., by  $\mathbb{R}$  (see [2] for the details). Besides roles, the logic makes use of *features*. Each feature has an associated concrete domain  $\mathcal{D}$  and represents a (functional) relation between objects and values of  $\mathcal{D}$ .

Starting from a set of concept names (denoted by the letter  $A$ ), a set of role names (denoted by  $R$ ), a set of unary predicate names (denoted by  $p$ ), and a set of features (denoted by  $f$ ), we inductively define the set of *concepts* (denoted by  $C$ ) as follows. Every concept name  $A$  is a concept (atomic concept), and for  $C_1$  and  $C_2$  concepts,  $R$  a role name,  $f$  a feature with associated domain  $\mathcal{D}$ , and  $p$  a unary predicate of  $\mathcal{D}$ , the following are concepts:

- $C_1 \sqcap C_2$  (conjunction),  $C_1 \sqcup C_2$  (disjunction), and  $\neg C$  (negation);
- $\exists R.C$  (existential restriction) and  $\forall R.C$  (universal restriction);
- $p(f)$  (predicate restriction).

To express intentional knowledge about concepts, we make use of a *concept hierarchy*, which is a set of assertions of the form  $A_1 \sqsubseteq A_2$  and  $A_1 \sqsubseteq \neg A_2$ , with  $A_1$  and  $A_2$  concept names. The former assertion expresses an *inclusion*, while the latter expresses a *disjointness*. For example,  $\text{football} \sqsubseteq \text{sport}$  and  $\text{male} \sqsubseteq \neg \text{female}$  could be assertions that are part of a concept hierarchy.

Formally, the semantics of concepts is defined by an *interpretation*  $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ , consisting of an *abstract domain*  $\Delta^{\mathcal{I}}$  and an *interpretation function*  $\cdot^{\mathcal{I}}$  that assigns to each concept name  $A$  a subset  $A^{\mathcal{I}}$  of  $\Delta^{\mathcal{I}}$ ; to each role name  $R$  a binary relation  $R^{\mathcal{I}}$  over  $\Delta^{\mathcal{I}}$ , and to each feature name  $f$ , associated with the concrete domain  $\mathcal{D}$ , a partial function  $f^{\mathcal{I}} : \Delta^{\mathcal{I}} \rightarrow \mathcal{D}$ . The interpretation function can be extended to arbitrary concepts as follows:

$$(C_1 \sqcap C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$$

$$\begin{aligned}
(C_1 \sqcup C_2)^{\mathcal{I}} &= C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} &= \neg C^{\mathcal{I}} \\
(\exists R.C)^{\mathcal{I}} &= \{c \in \Delta^{\mathcal{I}} \mid \text{there exists } d \in \Delta^{\mathcal{I}} \text{ s.t. } (c, d) \in R^{\mathcal{I}} \text{ and } d \in C^{\mathcal{I}}\} \\
(\forall R.C)^{\mathcal{I}} &= \{c \in \Delta^{\mathcal{I}} \mid \text{for all } d \in \Delta^{\mathcal{I}} \text{ s.t. } (c, d) \in R^{\mathcal{I}} \text{ we have } d \in C^{\mathcal{I}}\} \\
(p(f))^{\mathcal{I}} &= \{c \in \Delta^{\mathcal{I}} \mid f^{\mathcal{I}}(c) \in p^{\mathcal{D}}\}
\end{aligned}$$

An assertion  $A_1 \sqsubseteq A_2$  is *satisfied* by an interpretation  $\mathcal{I}$  if  $A_1^{\mathcal{I}} \subseteq A_2^{\mathcal{I}}$ . An assertion  $A_1 \sqsubseteq \neg A_2$  is satisfied by an interpretation  $\mathcal{I}$  if  $A_1^{\mathcal{I}} \cap A_2^{\mathcal{I}} = \emptyset$ . We call an interpretation that satisfies all assertions in a hierarchy  $\mathcal{H}$  a *model* of  $\mathcal{H}$ . A concept  $C$  is *satisfiable* in  $\mathcal{H}$  if  $\mathcal{H}$  admits a model  $\mathcal{I}$  such that  $C^{\mathcal{I}} \neq \emptyset$ . A hierarchy  $\mathcal{H}$  *logically implies* an assertion  $C_1 \sqsubseteq C_2$  between arbitrary concepts  $C_1$  and  $C_2$  if  $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$ , for each model  $\mathcal{I}$  of  $\mathcal{H}$ .

### 3 Representing User Profiles

We describe how to represent user profiles using the Description Logic presented in Section 2. The user profiles are tailored for dating services, though the same framework can be used, with small modifications, for different applications. We do not use the full expressive power of the Description Logic. In particular, we use a single role `hasInterest`, to express interest in topics<sup>1</sup>, and we make a limited use of the constructs. We assume the set of features to represent physical characteristics such as age, height, etc. Additionally, we use a special feature `level` that expresses the level of interest in a certain field. The concrete domain associated to `level` is the interval  $\{\ell \in \mathbb{R} \mid 0 < \ell \leq 1\}$ .

A user profile  $P$  consists of the conjunction of the following parts:

- A conjunction of *atomic concepts*, to represent atomic properties associated to the user. We denote the set of such concepts as  $Names(P)$ .
- A conjunction of concepts of the form  $p(f)$ , to represent physical characteristics. The (unary) predicate  $p$  can be one of the predicates  $\geq_{\ell}(\cdot)$ ,  $\leq_{\ell}(\cdot)$ ,  $=_{\ell}(\cdot)$ , where  $\ell$  is a value of the concrete domain associated to  $f$ , or any logical conjunction of them. We denote the set of such concepts as  $Features(P)$ . Since  $(p_1 \wedge p_2)(f)$  is equivalent to  $p_1(f) \sqcap p_2(f)$ , in the following, we can assume w.l.o.g. that  $Features(P)$  contains at most one concept of the form  $p(f)$  for each feature  $f$ .
- A conjunction of concepts of the form  $\exists hasInterest.(C \sqcap \geq_x(level))$ , where  $C$  is a conjunction of concept names, and  $0 \leq x \leq 1$ . Each such concept represents an interest in a concept  $C$  with level at least  $x$ . We denote the set of such concepts as  $Interests(P)$ .
- A conjunction of concepts of the form  $\forall hasInterest.(\neg C \sqcup \leq_x(level))$ , where  $C$  is a conjunction of concept names, and  $0 \leq x \leq 1$ . Each such concept represents the

<sup>1</sup>For modeling profiles in different contexts, additional roles could be added to this language. For example, `hasSkill` for expressing skills in certain fields.

fact that the interest in a concept  $C$  has level at most  $x$ . Note that, to represent the complete lack of interest in  $C$ , it is sufficient to put  $x = 0$ . We denote the set of such concepts as  $NoInterests(P)$ .

**Example 1** A supplied profile describing, say, a 35-years-old male, 1.82 cm tall, with strong interests in fantasy novels and japanese comics, fair interest in politics and no interest in football, could be expressed as follows:

$$\begin{aligned} & \text{male} \sqcap =_{35}(\text{age}) \sqcap =_{1.82}(\text{height}) \sqcap \\ & \exists \text{hasInterest} . (\text{fantasyNovels} \sqcap \geq_{0.8}(\text{level})) \sqcap \\ & \exists \text{hasInterest} . (\text{japaneseComics} \sqcap \geq_{0.8}(\text{level})) \sqcap \\ & \exists \text{hasInterest} . (\text{politics} \sqcap \geq_{0.4}(\text{level})) \sqcap \\ & \forall \text{hasInterest} . (\neg \text{football} \sqcup \leq_0(\text{level})) \end{aligned}$$

where we suppose that interests are organized in a hierarchy including  $\text{fantasyNovels} \sqsubseteq \text{novels}$ ,  $\text{japaneseComics} \sqsubseteq \text{comics}$ , and  $\text{male} \sqsubseteq \neg \text{female}$

Observe that, when a profile is demanded, usually features like **age** and **height** will be used with range predicates (e.g.,  $(\geq 30 \wedge \leq 70)(\text{age})$ ), instead of equality predicates as in the above example.

The following property follows immediately from the semantics of existential restriction. For every pair of concepts  $C_1$  and  $C_2$ , role  $R$ , feature  $f$  with associated concrete domain  $\mathcal{D}$ , and  $p$  a predicate of  $\mathcal{D}$ :

$$\text{if } \mathcal{H} \models C_1 \sqsubseteq C_2 \quad \text{then} \quad \mathcal{H} \models \exists R . (C_1 \sqcap \geq_\ell(f)) \sqsubseteq \exists R . (C_2 \sqcap \geq_\ell(f))$$

For example, if  $\text{football} \sqsubseteq \text{sport}$ , then someone with a level of interest  $\ell$  in **football** has at least the same level of interest in **sport**. This property is exploited in the matching algorithm provided in Section 4.

## 4 The Matching Algorithm

We present the algorithm for matching user profiles. The matching is performed over two profiles: the *demand* profile  $P_d$  and the *supply* profile  $P_s$ . The algorithm is not symmetric, i.e., it evaluates how  $P_s$  is suited for  $P_d$ , which is different from how  $P_d$  is suited for  $P_s$  [7]; of course, in order to determine how  $P_d$  is suited for  $P_s$ , we can simply exchange the arguments of the algorithm.

From a logical point of view, we extend the non-standard inferences *contraction* and *abduction* defined in [4]. In particular, our contraction either removes or weakens conjuncts from  $P_d$  so as to make  $P_d \sqcap P_s$  satisfiable in  $\mathcal{H}$ ; abduction, instead, either adds or strengthens conjuncts in  $P_s$  so as to make  $\mathcal{H} \models P_s \sqsubseteq P_d$ . The algorithm is based on structural algorithms for satisfiability and subsumption [3]. Since it is reasonable to assume that users do not enter contradicting information, we assume that the profiles  $P_d$  and  $P_s$  are consistent.

The result of the match is a *penalty* in  $\mathbb{R}$ : the larger the penalty, the less  $P_s$  is suited for  $P_d$ . In particular, partial penalties are added to the overall penalty by matching corresponding conjuncts of the two profiles; this is done in two ways.

**Contraction.** When a conjunct  $C_d$  in  $P_d$  is in contrast with some conjunct  $C_s$  in  $P_s$ , then  $C_d$  is removed and a penalty is added. Intuitively, since the supplier has something the demander does not like, in order to make the profiles match the demander *gives up* one of her requests. For example, let  $C_d = \forall \text{hasInterest}.\neg \text{sport} \sqcup \leq_{0.2}(\text{level})$  and  $C_s = \exists \text{hasInterest}.\text{football} \sqcap \geq_{0.4}(\text{level})$ , where we have  $\text{football} \sqsubseteq \text{sport}$  in  $\mathcal{H}$ . In this case the demander looks for someone who does not like sports very much, while the supplier likes football and therefore he likes sports. In this case, pursuing the match would require the demander to give up his/her request about sports, so the algorithm adds a penalty  $\Pi_{c\ell}(0.4, 0.2)$  that depends on the gap between the lower bound (0.4) of the supply and the upper bound (0.2) of the demand. Similarly, for a feature  $f$  with contrasting predicates  $p_d$  and  $p_s$ , a penalty  $\Pi_{cf}(p_d(f), p_s(f))$  is added to take into account the removal of  $p_d(f)$  from  $P_d$ . In case a concept  $A_d$  representing an atomic property has to be removed, the algorithm makes use of another penalty function  $\Pi_c(\cdot)$ , whose argument is the concept  $A_d$ .

**Abduction.** When a conjunct  $c_d$  in  $P_d$  has no corresponding conjunct in  $P_s$ , we add a suitable conjunct  $c_s$  in  $P_s$  that makes the profiles match, and add a corresponding penalty. Intuitively, the demander wants something which the supplier does not provide explicitly; in this case we assume that the supplier may or may not satisfy the demander's request, and as a consequence of this possibility of conflict we add a penalty. This is done by means of a penalty function  $\Pi_a(\cdot)$ , whose argument is a concept  $C$ , that takes into account the addition of  $C$  to  $P_s$ . When the level of interest must be strengthened, we use a function  $\Pi_{a\ell}(\cdot)$ , that takes into account the gap between bounds. Similarly, a penalty function  $\Pi_{af}(\cdot)$  takes into account the addition of features.

**Algorithm** CalculatePenalty

**Input** demand profile  $P_d$ , supply profile  $P_s$ , concept hierarchy  $\mathcal{H}$

**Output** real value penalty  $\geq 0$

penalty := 0;

// **Contraction**

**foreach**  $A_d \in \text{Names}(P_d)$  **do**

**if** there exists  $A_s \in \text{Names}(P_s)$

    such that  $\mathcal{H} \models A_d \sqsubseteq \neg A_s$

**then** remove  $A_d$  from  $P_d$

    penalty := penalty +  $\Pi_c(A_d)$

**foreach**  $p_d(f) \in \text{Features}(P_d)$  **do**

**if** there exists  $p_s(f) \in \text{Features}(P_s)$

    such that  $\exists x.p_d(x) \wedge p_s(x)$  is unsatisfiable in the domain associated to  $f$

**then** remove  $p_d(f)$  from  $P_d$

    penalty := penalty +  $\Pi_{cf}(p_d(f), p_s(f))$

**foreach**  $\exists \text{hasInterest}.(C_d \sqcap \geq_{x_d}(\text{level})) \in \text{Interests}(P_d)$  **do**

**foreach**  $\forall \text{hasInterest}.\neg C_s \sqcup \leq_{x_s}(\text{level}) \in \text{NoInterests}(P_s)$  **do**

**if**  $\mathcal{H} \models C_d \sqsubseteq C_s$  **and**  $x_d \geq x_s$

```

    then replace  $\exists\text{hasInterest}.(C_d \sqcap \geq_{x_d}(\text{level}))$  in  $P_d$ 
        with  $\exists\text{hasInterest}.(C_d \sqcap \geq_{x_s}(\text{level}))$ 
        penalty := penalty +  $\Pi_{cl}(x_d, x_s)$ 
foreach  $\forall\text{hasInterest}.(¬C_d \sqcup \leq_{x_d}(\text{level})) \in \text{NoInterests}(P_d)$  do
    foreach  $\exists\text{hasInterest}.(C_s \sqcap \geq_{x_s}(\text{level})) \in \text{Interests}(P_s)$  do
        if  $\mathcal{H} \models C_s \sqsubseteq C_d$  and  $x_d \leq x_s$ 
        then replace  $\forall\text{hasInterest}.(¬C_d \sqcup \leq_{x_d}(\text{level}))$  in  $P_d$ 
            with  $\forall\text{hasInterest}.(¬C_d \sqcup \leq_{x_s}(\text{level}))$ 
            penalty := penalty +  $\Pi_{cl}(x_s, x_d)$ 
// Abduction
foreach  $A_d \in \text{Names}(P_d)$  do
    if there does not exist  $A_s \in \text{Names}(P_s)$  such that  $\mathcal{H} \models A_s \sqsubseteq A_d$ 
    then add  $A_d$  to  $P_s$ 
        penalty := penalty +  $\Pi_a(A_d)$ 
foreach  $p_d(f) \in \text{Features}(P_d)$  do
    if there exist  $p_s(f) \in \text{Features}(P_s)$ 
    then if  $\forall x.p_s(x) \Rightarrow p_d(x)$  is false in the domain associated to  $f$ 
        then add  $p_d(f)$  to  $P_s$ 
            penalty := penalty +  $\Pi_{af}(p_d(f), p_s(f))$ 
        else add  $p_d(f)$  to  $P_s$ 
            penalty := penalty +  $\Pi_{af}(p_d(f), \top(f))$ 
foreach  $\exists\text{hasInterest}.(C_d \sqcap \geq_{x_d}(\text{level})) \in \text{Interests}(P_d)$  do
    if there does not exist  $\exists\text{hasInterest}.(C_s \sqcap \geq_{x_s}(\text{level})) \in \text{Interests}(P_s)$ 
        such that  $\mathcal{H} \models C_s \sqsubseteq C_d$  and  $x_s \geq x_d$ 
    then if there exists  $\exists\text{hasInterest}.(C_s \sqcap \geq_{x_s}(\text{level})) \in \text{Interests}(P_s)$ 
        such that  $\mathcal{H} \models C_s \sqsubseteq C_d$ 
        then let  $\exists\text{hasInterest}.(C_s \sqcap \geq_{x_s}(\text{level}))$  be the concept in  $\text{Interests}(P_s)$ 
            with maximum  $x_s$  among those for which  $\mathcal{H} \models C_s \sqsubseteq C_d$  holds
            penalty := penalty +  $\Pi_{al}(x_d, x_s)$ 
        else penalty := penalty +  $\Pi_a(\exists\text{hasInterest}.(C_d \sqcap \geq_{x_d}(\text{level})))$ 
        add  $\exists\text{hasInterest}.(C_d \sqcap \geq_{x_d}(\text{level}))$  to  $P_s$ 
foreach  $\forall\text{hasInterest}.(¬C_d \sqcup \leq_{x_d}(\text{level})) \in \text{NoInterests}(P_d)$ 
    if there does not exist  $\forall\text{hasInterest}.(¬C_s \sqcup \leq_{x_s}(\text{level})) \in \text{NoInterests}(P_s)$ 
        such that  $\mathcal{H} \models C_d \sqsubseteq C_s$  and  $x_d \geq x_s$ 
    then if there exists  $\forall\text{hasInterest}.(¬C_s \sqcup \leq_{x_s}(\text{level})) \in \text{NoInterests}(P_s)$ 
        such that  $\mathcal{H} \models C_d \sqsubseteq C_s$ 
        then let  $\forall\text{hasInterest}.(¬C_s \sqcup \leq_{x_s}(\text{level}))$  be the concept in  $\text{Interests}(P_s)$ 
            with minimum  $x_s$  among those for which  $\mathcal{H} \models C_d \sqsubseteq C_s$  holds
            penalty := penalty +  $\Pi_{al}(x_s, x_d)$ 
        else penalty := penalty +  $\Pi_a(\forall\text{hasInterest}.(¬C_d \sqcup \leq_{x_d}(\text{level})))$ 
        add  $\forall\text{hasInterest}.(¬C_d \sqcup \leq_{x_d}(\text{level}))$  to  $P_s$ 
return penalty

```



The penalty functions used in the algorithm are defined as follows.

- For an atomic concept  $A_d$ ,  $\Pi_c(A_d)$  and  $\Pi_a(A_d)$  depend solely from domain knowledge; for example, if the demander searches for a female while the supplier is a male, we are expected to associate a very high penalty to  $\Pi_c(\text{female})$  while removing female from  $P_d$  in the contraction phase.
- Given a feature  $f$  and the predicates  $p_d(f)$ , and  $p_s(f)$ , let  $I_d$  and  $I_s$  be the intervals associated to  $p_d$  and  $p_s$  respectively, and  $G$  the gap between them; we define

$$\Pi_{cf}(p_d(f), p_s(f)) = \frac{|G|}{|I_d \cup I_s \cup G|}$$

In other words, the penalty is calculated by dividing the gap between  $I_d$  and  $I_s$  by the sum of the sizes of  $I_d$ ,  $I_s$ , and  $G$ .

For abduction we define (notice that, since  $P_d^c \sqcap P_s$  is consistent, there is no gap  $G$ , and since  $\forall x.p_s(x) \Rightarrow p_d(x)$  is false in the domain associated to  $f$ , we have that  $|I_s| > 0$ ):

$$\Pi_{af}(p_d(f), p_s(f)) = \frac{|I_s \setminus I_d|}{|I_s|}$$

- Given  $x_d, x_s \in [0, 1]$ ,  $\Pi_{cl}(x_d, x_s) = x_d - x_s$  and  $\Pi_{al}(x_d, x_s) = \frac{x_d - x_s}{1 - x_s}$ .
- For  $C_d = \sqcap_{i=1}^n A_i$ , we define

$$\Pi_a(\exists \text{hasInterest.}(C_d \sqcap \geq_{x_d}(\text{level}))) = x_d \cdot \sum_{i=1}^n \Pi_a(A_i)$$

$$\Pi_a(\forall \text{hasInterest.}(\neg C_d \sqcup \leq_{x_d}(\text{level}))) = \frac{1 - x_d}{\sum_{i=1}^n \frac{1}{\Pi_a(A_i)}}$$

Note that only the penalty functions  $\Pi_a(\cdot)$  and  $\Pi_c(\cdot)$ , when calculated on atomic concepts, rely on domain knowledge; all other penalty functions are defined based on the previous ones, and independently of other domain knowledge.

It is easy to check that all subsumption tests  $\mathcal{H} \models C_1 \sqsubseteq C_2$  in the algorithm can be done in polynomial time in the size of  $\mathcal{H}$ ,  $C_1$ , and  $C_2$ . Hence, it can be straightforwardly proved that the complexity of the algorithm is polynomial w.r.t. the size of the input.

$$\begin{array}{ll} P_d = \text{male} \sqcap >_{30}(\text{age}) \sqcap >_{1.80}(\text{height}) & P_s = \text{male} \sqcap =_{35}(\text{age}) \sqcap =_{1.70}(\text{height}) \\ \sqcap \exists \text{hasInterest.}(\text{literature} \sqcap \geq_{0.5}(\text{level})) & \sqcap \exists \text{hasInterest.}(\text{fantasyNovels} \sqcap \geq_{0.8}(\text{level})) \\ \sqcap \exists \text{hasInterest.}(\text{politics} \sqcap \geq_{0.4}(\text{level})) & \sqcap \exists \text{hasInterest.}(\text{japaneseComics} \sqcap \geq_{0.8}(\text{level})) \\ & \sqcap \forall \text{hasInterest.}(\neg \text{football} \sqcup \leq_0(\text{level})) \end{array}$$

Figure 1: Formalization of profiles of Example 2

**Example 2** Let  $P_d$  be the demand for a "man over thirty, taller than 180 cm, with fair interest in literature and politics" and  $P_s$  the supplied profile describing a "35 year-old male, 1.70 cm tall, with strong interest in fantasy novels and japanese comics and no interest in football". Such profiles are formalized in Figure 1 w.r.t. a hierarchy  $\mathcal{H}$  including  $\text{fantasyNovels} \sqsubseteq \text{novels}$ ,  $\text{japaneseComics} \sqsubseteq \text{comics}$ ,  $\text{comics} \sqsubseteq \text{literature}$  and  $\text{novels} \sqsubseteq \text{literature}$ . The evaluation of the matching algorithm on  $P_s$  and  $P_d$  w.r.t.  $\mathcal{H}$  returns a penalty value equal to  $\Pi_{cf}(1.80, 1.70) + \Pi_a(\exists \text{hasInterest}(\text{politics} \sqcap \geq_{0.4}(\text{level})))$ . The first term represents the need of giving up the height requirement in  $P_d$  during the contraction phase, while the second one takes into account the addition of politics among  $\text{Interests}(P_s)$  during the abduction phase.

The following theorem establishes the correctness of the above algorithm w.r.t. the computation of contraction and abduction. We denote with  $P_d^c$  the profile  $P_d$  after contraction, and with  $P_s^a$  the profile  $P_s$  after abduction.

**Theorem 3** *Given a concept hierarchy  $\mathcal{H}$ , a demand profile  $P_d$ , and a supply profile  $P_s$ , the following properties hold: (i)  $P_d^c \sqcap P_s$  is satisfiable in  $\mathcal{H}$ ; (ii)  $P_s^a$  is satisfiable in  $\mathcal{H}$ ; (iii)  $\mathcal{H} \models P_d^c \sqsubseteq P_s^a$ . (iv) there does not exist a profile  $P'_s$  more general than  $P_s^a$  (i.e.,  $\mathcal{H} \models P_s^a \sqsubseteq P'_s$  and  $\mathcal{H} \not\models P'_s \sqsubseteq P_s$ ) such that  $\mathcal{H} \models P'_s \sqsubseteq P_s$  and  $\mathcal{H} \models P'_s \sqsubseteq P_d^c$ .*

*Proof (sketch).* (i) The proof is by construction of a model  $\mathcal{I}$  of  $\mathcal{H}$  such that  $(P_d^c \sqcap P_s)^{\mathcal{I}} \neq \emptyset$ . (ii) Follows directly from (i), since in the abduction step we add to  $P_s$  conjuncts that are already in  $P_d^c$ . (iii) and (iv) Follow by construction of  $P_s^a$ , since exactly those conjuncts of  $P_d^c$  that are not subsumed by  $P_s$  in  $\mathcal{H}$  have been included in  $P_s^a$ . By the fact that  $\mathcal{H}$  consists only of inclusions and disjointness assertions between pairs of atomic concepts, it is indeed sufficient to consider pairs of concepts to check subsumption.  $\square$

## 5 Conclusions

In this paper we have addressed the problem of matching user profiles, when the demander's and supplier's profiles can have missing or conflicting information. In such a case, we have to take into account that the demander may need to give up some of her requests, and/or she may need to make assumptions on unspecified properties of the supplier's profile. We have proposed a DL-based framework for expressing user profiles in this setting, and a language suited for dating services. We have proposed an ad-hoc structural algorithm for matching profiles that, given a demander's and a supplier's profile, returns a penalty: the higher the penalty, the less the two profiles are compatible. As a future work, we want to test the algorithm in real cases with a prototype that is currently under development: we believe that promising applications of our techniques can be dating, recruitment, and service discovery systems.

**Acknowledgments** The first two authors were partly supported by MIUR under FIRB (Fondo per gli Investimenti della Ricerca di Base) project "MAIS: Multichannel Adaptive Information Systems" in the context of the Workpackage 2 activities.

## References

- [1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.
- [2] F. Baader and P. Hanschke. A schema for integrating concrete domains into concept languages. In *Proc. of IJCAI'91*, pages 452–457, 1991.
- [3] A. Borgida and P. F. Patel-Schneider. A semantics and complete algorithm for subsumption in the CLASSIC description logic. *J. of Artificial Intelligence Research*, 1:277–308, 1994.
- [4] S. Colucci, T. Di Noia, E. Di Sciascio, F. M. Donini, and M. Mongiello. Concept abduction and contraction in description logics. In *Proc. of DL 2003*. CEUR Electronic Workshop Proceedings, <http://ceur-ws.org/Vol-81/>, 2003.
- [5] S. Colucci, T. Di Noia, E. Di Sciascio, F. M. Donini, M. Mongiello, and M. Motola. A formal approach to ontology-based semantic match of skills descriptions. *J. of Universal Computer Science*, Special issue on Skills Management, 2003.
- [6] T. Di Noia, E. Di Sciascio, F. M. Donini, and M. Mongiello. Abductive match-making using description logics. In *Proc. of IJCAI 2003*, pages 337–342, 2003.
- [7] T. Di Noia, E. Di Sciascio, F. M. Donini, and M. Mongiello. A system for principled matchmaking in an electronic marketplace. In *Proc. of WWW 2003*, pages 321–330, May 20–24 2003.
- [8] Z. Galil. Efficient algorithms for finding maximum matching in graphs. *ACM Computing Surveys*, 18(1):23–38, 1986.
- [9] F. S. Hillier and G. J. Lieberman. *Introduction to Operations Research*. McGraw-Hill, 1995.
- [10] J. Kennington and Z. Wang. An empirical analysis of the dense assignment problem: Sequential and parallel implementations. *ORSA Journal on Computing*, 3(4):299–306, 1991.
- [11] D. Kuokka and L. Harada. Integrating information via matchmaking. *J. of Intelligent Information Systems*, 6:261–279, 1996.
- [12] 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.
- [13] D. Veit, J. P. Müller, M. Schneider, and B. Fiehn. Matchmaking for autonomous agents in electronic marketplaces. In *Proc. of AGENTS '01*, pages 65–66. ACM, 2001.