Automatic Identity Recognition in the Semantic Web [*]

Alfio Ferrara, Davide Lorusso, Stefano Montanelli

Università degli Studi di Milano,
DICo, 10235 Milano, Italy,
{ferrara, lorusso, montanelli}@dico.unimi.it

Abstract. The OKKAM initiative[1] has recently highlighted the need of moving from the traditional web towards a "web of entities", where real-world objects descriptions could be retrieved, univocally identified, and shared over the web. In this paper, we propose our vision of the entity recognition problem and, in particular, we propose methods and techniques to capture the "identity" of a real entity in the Semantic Web. We claim that automatic techniques are needed to compare different RDF descriptions of a domain with the goal of automatically detect heterogeneous descriptions of the same real-world objects. Problems and techniques to solve them are discussed together with some experimental results on a real case study on web data.

1 Introduction

The Semantic Web is a web of data[2], but, we could add, it is (or should be) also a web on entities. In the Semantic Web context, people and organizations provide a explicit and semantically rich descriptions of their contents on the web, enabling human and machine interaction as well as data integration. In doing this, concrete and abstract real-world entities (e.g., persons, products, locations) are searchable on the web by exploiting their RDF descriptions. But, obviously, people perceive and describe the same real entity in many different ways. The risk of this is to produce the so-called "archipelago of semantic islands" [1], where each description of the same entity is different form the others and where data and information cannot be integrated and re-used. In other terms, what we loose here is not the real entity, which is available in many documents in many ways, but rather the "identity" of the real-entity. Now, what does it mean really the word "identity"? Despite the philosophical issues behind the question, we could agree on the fact that the identity is the way we use in order i) to refer to an object, ii) to distinguish an object from another one, and iii) to collect and integrate information about an object. This problem has been solved in the

[*] This paper has been partially funded by the BOEMIE Project, FP6-027538, 6th EU Framework Programme.

[1] http://www.okkam.org/.

[2] http://www.w3.org/2001/sw/.

traditional web for what concerns resources and documents by using the URI standard. The question here is if we can provide something similar to URIs for entities. If we want to capture the "identity" of a real entity in the Semantic Web, we need to be able to compare different RDF descriptions of a domain with the goal of automatically detect descriptions referred to the same real-world object. We call this problem "Identity Recognition Problem". In this paper, we discuss the identity recognition problem by focusing on three sub-problems and by proposing specific methods and techniques to solve them. Moreover, we present a case study where these techniques are shown in action and some experimental results are provided by using the instance matching functionalities implemented in our ontology matching tool HMatch, which is developed in the framework of the BOEMIE project [2,3].

2 The Identity Recognition Problem

We call *individual* any abstract or concrete entity in a domain of interest. When a domain is described by a formal representation of its contents, we do not have a direct knowledge of the individuals *per se*, but only a set of *descriptions* or *perceptions* of these individuals. A domain representation is seen as a set of descriptions. A description is an arbitrary set of assertions about domain objects and data values specified according to a formal language and a data model. In this respect, the problem of individual identification is defined as:

1. the problem of detecting, out of the several assertions that characterize a description, those assertions which denote a specific individual in the domain;
2. the problem of detecting, out of several descriptions in a domain representation, those descriptions which denote the same individual in the domain.

A domain representation is provided according to a formal language and a data model. Data provided in the model can be typed and featured by a well defined semantics (e.g., DL ontologies), structured by a relational or object-oriented model (e.g., relational or object-relational databases), or organized as potentially untyped graphs (e.g., RDF, RDFS). We assume to generalize domain representations as RDF untyped graphs, where nodes denote objects or atomic data and edges denote labeled binary relations. To give an example of individual representations, let us suppose to represent a DVD collection. Two examples of RDF individual descriptions are the following ones:

Description 1	**Description 2**
⟨#d01⟩ ⟨#title⟩ scarface	⟨#d01⟩ ⟨#title⟩ scarface
⟨#d01⟩ ⟨#director⟩ Brian De Palma	⟨#d01⟩ ⟨#director⟩ ⟨#d02⟩
⟨#d01⟩ ⟨#actor⟩ Al Pacino	⟨#d02⟩ ⟨#name⟩ Brian De Palma
⟨#d01⟩ ⟨#actor⟩ Michelle Pfeiffer	⟨#d01⟩ ⟨#actor⟩ ⟨#d03⟩
⟨#d01⟩ ⟨#price⟩ 14.99	⟨#d03⟩ ⟨#name⟩ Al Pacino
	⟨#d01⟩ ⟨#actor⟩ ⟨#d04⟩
	⟨#d03⟩ ⟨#name⟩ Michelle Pfeiffer
	⟨#d01⟩ ⟨#price⟩ 14.99

Intuitively, we have in both cases a description of a DVD (i.e., d01) and some data about it. But, how many different individuals are represented in the two descriptions? Which properties are referred to which individual? How can we detect individuals here?
We can maybe agree on the fact that, in both cases, five real individuals are represented: four concrete individuals (i.e., the DVD, the persons Brian De Palma, Al Pacino and Michelle Pfeiffer) and an abstract individual (i.e., the movie "Scarface"). But we have different properties in the two cases: title is an attribute of the DVD or an attribute of the movie "Scarface", or both? Moreover, persons have a name in the second representation, but no attributes in the first. Finally, supposing to have a correct identification of individuals in the two cases, how are we supposed to match these two different representations?
We can try to address these questions by splitting the Identity Recognition Problem into three sub-problems:

1. Individuals identity and essence: what is an individual and how it is identified?
2. Individuals description: how to compare two different descriptions of the same individual?
3. Individuals existence: how to deal with the fact that some individuals descriptions "contain" descriptions of other individuals? When an individual has an independent existence?

In the following sections we try to formalize these problems and sketch possible techniques to address them.

2.1 A Real Case Study

In order to test and discuss the techniques proposed in this paper over real data, we created a case study using data about DVDs provided by the Amazon.com web service[3] together with data provided about movies by the IMDB database[4]. The goal of our test case is to identify movies out of these datasources. For what concerns Amazon, movie identification is not trivial, because data describe DVD editions of the movies and not the movies per se. Thus, we want to cluster together DVDs referring to the same movie and to extract out of DVD descriptions the movie description (i.e., *extraction*). In our ideal case, a user who is going to create a RDF movie collection would be able to get a reference to "Scarface" by automatically exploring the Amazon DVD collection. On the other side, IMDB provides data about movies. In such a case, the goal is to compare IMDB movie descriptions with movie descriptions extracted from Amazon, in order to automatically identify different descriptions referred to the same real movie (i.e., *mapping*). These two goals are graphically presented in Figure 1.

[3] http://aws.amazon.com/

[4] http://imdb.com/. Despite the fact that IMDB is an Amazon.com company, data provided by the two services are very different, at least with respect to the user query.

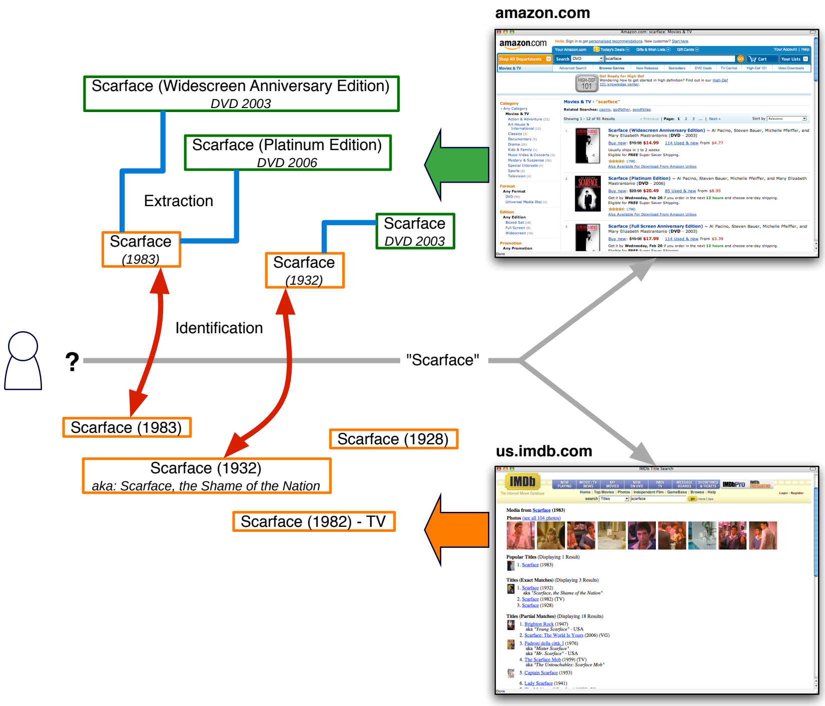

Fig. 1. Graphical representation of entity extraction and mapping

In order to evaluate our techniques, we manually built a set of expected results. From Amazon, we got data resulting for the query "Scarface" over the DVDs collection. Results contain 87 items (describing videos and DVDs). Manually, we categorized these items with respect to the related movie: we obtained 15 DVD editions of the movie "Scarface (1983)", 4 of the movie "Scarface (1932)", 6 DVD collections including the movie "Scarface (1983)", 11 of the movie "Mr. Scarface (1976)", 4 of the movie "Captain Scarface (1953)" and 47 DVD editions of other movies/materials not directly related to the query "Scarface". Concerning IMDB, we collected 27 movies, including the ones in Amazon and others. Finally, we imported data into two very simple RDF schemas, with the goal of preserving as much as possible the original structure of data provided by the two resources. A portion of resulting RDF is shown in Figure 2.

3 Identity of Individuals

In general, when looking at a real individual we focus on some of its properties in order to distinguish it from other individuals. However, not all the individual properties can be used to perform such a distinction. In the Aristotelian theory of definition, some properties are considered as part of the individual essence, some

Amazon item	**IMDB movie**
⟨#item-2⟩ ⟨#title⟩ Scarface, Special Edition	⟨#M-01⟩ ⟨#title⟩ Scarface (1983)
⟨#item-2⟩ ⟨#director⟩ Brian De Palma	⟨#M-01⟩ ⟨#director⟩ ⟨#P04⟩
⟨#item-2⟩ ⟨#actor⟩ Al Pacino	⟨#M-01⟩ ⟨#cast⟩ ⟨#C06⟩
⟨#item-2⟩ ⟨#format⟩ PAL	⟨#P04⟩ ⟨#name⟩ Brian De Palma
⟨#item-2⟩ ⟨#hasLanguage⟩ ⟨#L06⟩	⟨#P04⟩ ⟨#birth⟩ 11-09-1940
⟨#L06⟩ ⟨#name⟩ Spanish	⟨#C06⟩ ⟨#actor⟩ ⟨#P16⟩
	⟨#C06⟩ ⟨#character⟩ Tony Montana
	⟨#P16⟩ ⟨#name⟩ Al Pacino

Fig. 2. A portion of RDF created from Amazon and IMDB data

other are just "accidental" attributes. Borrowing this intuition, we can assume that, given an individual description $D_o = \{\mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_n\}$ as a set of assertions about an individual o, the identification I^o of o is an identity function $I^o(D_o) \rightarrow \{(d^i, v^i)\}$ which associates to each group of assertions in D_o an identification strength v^i in the range [0,1]. Goal of the identification function is to provide a measure of the importance of each group of assertions with respect to the general goal of detecting the identity of o.

Definition 1. *Identity Function. Given an individual description $D = \{\mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_n\}$, the Identity Function $I(D) \rightarrow \{(d^i, v^i)\}$ is defined as follows:*

$$\forall d^i \in \mathcal{P}(D),\ d^i \rightarrow v^i$$

where $\mathcal{P}(D)$ denotes the power set of D, d^i denotes a possible subset of assertions in D, while v^i is a measure (in the range [0,1]) of the importance of d^i for the identification of o.

3.1 Implementation of the Identity Function

The identity function can be obtained in three different ways:

1. by taking into account schema constraints when available (e.g., functional properties in DL, key constraints in relational databases);
2. by taking into account domain specific information or human suggestions (e.g., ISBN codes for books, personal identification numbers for persons, manual configuration of an identity recognition system);
3. by exploiting statistical techniques over data values. Usually, the idea behind these techniques is that assertions with a high number of different values are more useful for identification than assertions with a high number of similar or equal values.

In HMatch, identifying properties can be manually chosen or automatically detected by taking into account, for each group of assertions $d \in D$, the number $N^d_{\neq}$ of different values retrieved for d in all the individuals of the same type of the given one. More formally, given an individual o, we take into account the set

O of all the individuals of the same type of o in the RDF description. Then, we calculate the identity function as follows:

$$\forall d^i \in \mathcal{P}(D),\ d^i \rightarrow \frac{N_{\neq}^{d^i}}{\mid O_P \mid}$$

where $N_{\neq}^{d^i}$ is the number of different values of the assertions d^i over the set O, while $\mid O_P \mid$ is the number of different properties featuring the individuals in O.

Example. To see how the function works, let us suppose to have three simple movie descriptions:

Description 1	**Description 2**	**Description 3**
⟨#M01⟩ ⟨#title⟩ Scarface ⟨#M01⟩ ⟨#year⟩ 1983 ⟨#M01⟩ ⟨#genre⟩ Crime	⟨#M02⟩ ⟨#title⟩ Scarface ⟨#M02⟩ ⟨#year⟩ 1932	⟨#M03⟩ ⟨#title⟩ Testament ⟨#M03⟩ ⟨#year⟩ 1983

By taking into account all possible subsets of movie properties, the results of the identification function is shown in Table 1.

Table 1. Example of results provided by the identification function

Property subset	**Num. of different values**	**Num. of properties**	**Result**
{#title}	2	3	0.67
{#year}	2	3	0.67
{#genre}	1	3	0.34
{#title, #year}	3	3	1.0
{#title, #genre}	2	3	0.67
{#year, #genre}	2	3	0.67
{#title, #year, #genre}	3	3	1.0

Looking at the results, we can see how the function does not take into account properties with missing values, because they do not provide useful information. Moreover, the title or the year are not useful if we consider them separated, but they are enough to identify movies when taken into account in combination.

4 Heterogeneous Individual Descriptions

The same individual can be described in many different ways. Descriptions of the same individual can be different for several reasons, including the use of different languages or data models and a different conceptualization of the same domain. The use of a generalized RDF model does not avoid the presence of heterogeneous descriptions of the same individual. Thus, a key requirement in individual identification is the capability of comparing heterogeneous descriptions of the individuals in order to cluster together similar or matching descriptions. Given an individual o and two descriptions $D_o^1 = \{\mathcal{A}_0^1, \mathcal{A}_1^1, \ldots, \mathcal{A}_n^1\}$ and

$D^2_o = \{\mathcal{A}^2_0, \mathcal{A}^2_1, \ldots, \mathcal{A}^2_m\}$ of o, the identification I^o of o is a matching function $I^o(D^1_o, D^2_o) \rightarrow (D^{1,2}, v)$. Goal of the function $I^o(D^1_o, D^2_o)$ is to understand when two descriptions denote the same real individual in the domain.

Definition 2. *Matching Function. Given two individual descriptions D^1 and D^2, the matching function $I(D^1, D^2)$ is defined as follows:*

$$I(D^1, D^2) \rightarrow (D^{1,2}, v),$$

where v is a measure in the range [0,1] of the similarity of D^1 and D^2 and $D^{1,2}$ is the set of assertion mappings between D^1 and D^2. The goal of the matching function is to provide a measure of similarity between D^1 and D^2, under the assumption that the higher the descriptions similarity is, the higher is the probability that D^1 and D^2 denote the same real individual.

4.1 Implementation of the Matching Function

A typical implementation of the matching function is based on the idea of comparing the data values associated with a given individual. But, since different descriptions may have a different structure, how can we know which graph nodes in the two descriptions should be compared? The approach used in HMatch is to rely on the results provided by matching the two datasource schemas. In order to deal with highly different data structures, we need complex mappings between the concepts and properties of the two schemas. HMatch itself implements several different techniques for schema matching, including linguistic, structural and contextual matching [2]. Many other techniques have been proposed for schema matching [4]. Since schema matching is not the goal of the paper, we present the instance matching approach used in HMatch by assuming to have complex mappings at the schema level. HMatch evaluates the degree of similarity among different individuals by considering those assertions which provide a description of the individuals features. Consequently, the similarity of role filler values as well as the similarity of their direct types is evaluated. When two instances are compared, their similarity is proportional to the number of similar roles and role fillers they share. Moreover, for the similarity evaluation we use the identification power of properties provided by the identity function. The approach adopted in HMatch is based on the idea of considering properties as connections between individuals and propagating similarity values through them. Each specification of an individual is represented by means of a tree. In order to evaluate the degree of similarity of two individuals, the procedure computes a measure of similarity between datatype values and propagates these similarity degrees to the individuals of the higher level by combining the similarity among their property fillers. To this end, HMatch provides a set of specific techniques devoted to the evaluation of similarity between datatype values. A function called *datatype role filler matching* is responsible of selecting the most suitable matching technique for each pair of datatype role fillers, according to the semantic meaning of the roles and to the datatype category.

Example . Consider two individuals item-1 and M-01 representing different RDF descriptions of the movie *Scarface* (Figure 3). item-1 describes a DVD sold by *Amazon* while M-01 describes directly the movie.

Amazon item	**IMDB movie**
⟨#item-1⟩ ⟨#title⟩ Scarface [Region 2]	⟨#M-01⟩ ⟨#title⟩ Scarface (1983)
⟨#item-1⟩ ⟨#director⟩ Brian De Palma	⟨#M-01⟩ ⟨#director⟩ De Palma, Brian
⟨#item-1⟩ ⟨#actor⟩ Al Pacino	⟨#M-01⟩ ⟨#Year⟩ 1983
⟨#item-1⟩ ⟨#actor⟩ Michelle Pfeiffer	⟨#M-01⟩ ⟨#Country⟩ USA
⟨#item-1⟩ ⟨#theatricalReleaseDate⟩ 1983	⟨#M-01⟩ ⟨#cast⟩ ⟨#C06⟩
⟨#item-1⟩ ⟨#regionCode⟩ 2	⟨#C06⟩ ⟨#character⟩ Tony Montana
⟨#item-1⟩ ⟨#format⟩ NTSC,aspect ratio 2.35:1	⟨#C06⟩ ⟨#actor⟩ ⟨#P15⟩
	⟨#P15⟩ ⟨#name⟩ Pacino, Al
	⟨#M-01⟩ ⟨#cast⟩ ⟨#C13⟩
	⟨#C13⟩ ⟨#character⟩ Elvira Hancock
	⟨#C13⟩ ⟨#actor⟩ ⟨#P16⟩
	⟨#P16⟩ ⟨#name⟩ Pfeiffer, Michelle
	⟨#M-01⟩ ⟨#cast⟩ ⟨#C20⟩
	⟨#C20⟩ ⟨#character⟩ Manny Ribera
	⟨#C20⟩ ⟨#actor⟩ ⟨#P17⟩
	⟨#P17⟩ ⟨#name⟩ Bauer, Steven

Fig. 3. RDF description of item-1 and M-01

First of all we suppose to have the following set of mappings at schema level produced by a schema matching tool: $\{$(*title*,*title*), (*director*, *director*), (*actor*, *actor.name*), (*theatricalReleaseDate*, *Year*)$\}$. These mappings reduce the nested structure of the M-01 description to a flat one, making the M-01 description more comparable with the one of item-1. For each pair of properties involved in a mapping, HMatch compares the two values by means of a matching function specific for the category of the property (e.g. person name, date), which produces a similarity measure in the range [0,1]. The two property values are considered similar if the similarity measure is greater then a given threshold. For instance, the comparison of (Amazon) *director* and (IMBD) *director* property values, which are "Brian De Palma" and "De Palma, Brian" respectively, is performed with a matching function for person names and produces a similarity measure of 0.9. When multiple values are defined for the same properties, as for *actor* and *actor.name*, each possible value of the first property is compared with each possible value of the second. Then a set similarity measure is produced by considering similar values as the intersection of the two sets. When all the values of mapped property have been compared, the overall similarity measure of the two individuals is computed with the dice coefficient method. Assuming that, after all the comparisons, M-01 and item-1 have similar values for the properties (director, director), (actor, actor.name) and (theatricalReleaseDate, Year), but not for the properties (title,title), the overall similarity of the two individuals is evaluated as follows:

$$sim(\mathsf{item\text{-}1}, \mathsf{M\text{-}01}) = \frac{\text{similar properties}}{\text{total properties}} = \frac{6}{8} = 0.75$$

Having a reasonable threshold value of 0.6, the two individual descriptions are considered similar, which is correct since they are referred to the same movie.

5 Autonomous Identity

In the two previous sections, we assumed to work with individual descriptions. But, a description is just an arbitrary collection of assertions. So, the problem here is: how can we select the minimal set of assertions describing an individual? In other terms, we could also wondering if there is a method to identify an autonomous individual out of a domain representation. The most simple intuition here is to consider all the assertions having the same domain object as subject. This method can be easily generalized to cope with more nested graph structures, by taking into account all the assertions that are directly or indirectly connected with a domain object. However, this approach is not sufficient in many cases. This problem can be introduced by looking at the example of Figure 4. The two descriptions in the example represent two different DVD editions of the same movie. This means that there is a real individual (i.e., the movie "Scarface") which is not represented by a specific description. In general, a good design of the source schema should avoid this anomalies, but in real data it happens very often to have individuals that are "hidden" into descriptions of other individuals. For example, when talking about books, movies, music works and other kind of abstract individuals, it happens that the *work* is not distinguished from its physical edition. Also in other domains this is a problem: we have persons descriptions hidden into business transaction descriptions, geographical locations hidden into travel descriptions, and so on. How can we deal with this kind of individuals? In this context, the identity I^o of an hidden individual o is a discovery function $I^o(D) \rightarrow \{(d^i, v^i)\}$ which is defined as follows:

Description 1	**Description 2**
A1. ⟨#dvd_1⟩ ⟨#title⟩ Scarface	A2. ⟨#dvd_2⟩ ⟨#title⟩ Scarface
B1. ⟨#dvd_1⟩ ⟨#price⟩ 14.99	B2. ⟨#dvd_2⟩ ⟨#price⟩ 18.89
C1. ⟨#dvd_1⟩ ⟨#year⟩ 2003	C2. ⟨#dvd_2⟩ ⟨#year⟩ 2006
D1. ⟨#dvd_1⟩ ⟨#subtitles⟩ French, Spanish	D2. ⟨#dvd_2⟩ ⟨#subtitles⟩ French
E1. ⟨#dvd_1⟩ ⟨#director⟩ Brian De Palma	E2. ⟨#dvd_2⟩ ⟨#director⟩ Brian De Palma
F1. ⟨#dvd_1⟩ ⟨#actor⟩ Al Pacino	F2. ⟨#dvd_2⟩ ⟨#actor⟩ Al Pacino
G1. ⟨#dvd_1⟩ ⟨#actor⟩ Michelle Pfeiffer	G2. ⟨#dvd_2⟩ ⟨#actor⟩ Michelle Pfeiffer

Fig. 4. Example of the discovery problem

Definition 3. *Discovery Function. Given an arbitrary description D, the discovery function $I^o(D) \rightarrow (d^i, v^i)$ is defined as:*

$$\forall d^i \in \mathcal{P}(D),\ d^i \rightarrow v^i$$

where d^i denotes a possible subset of assertions of D and v^i (in the range [0,1]) denotes the probability of d^i to be the description of an hidden individual o.

5.1 Implementation of the Discovery Function

In HMatch, the discovery function is implemented using matching. The idea is to execute matching over the same datasource by collecting a log of the properties matching. Intuitively, in such a way we collect the properties that, in most of the cases, have equal or similar values in a set O of individuals. Of course, there are several properties whose values match in many cases even if they do not denote an autonomous individual. For example, the format of a DVD (e.g., PAL) is the same in many cases, even if it does not denote a movie. To deal with this situation, we work under the assumption that only a significant group of properties matching together in many cases are denoting an autonomous individual. We find all the subsets of properties matching and, for each subset d^i, we take into account the number of properties in the subset ($\mid d^i \mid$) and the number of occurrences of matching values v^i of the properties in the subset over O. Given the subset d_{max} as the subset containing the maximum number of properties, we associate with d^i the measure $\overline{d}^i = \frac{|d|}{|d_{max}|}$. Analogously, given the maximum number of matching values occurrences v_{max}, we associate with d^i the measure $\overline{v}^i = \frac{v^i}{v_{max}}$. As a result, given the subset of matching properties d^i, its measure of discovery $D(d^i)$ is calculated as:

$$D(d^i) = \overline{d}^i + \overline{v}^i.$$

In such a way, we consider the subset with the best balance between the number of properties and the number of matching values over its properties as the set of properties candidate to denote an autonomous individual description within a description of other individuals.

Example. As an example, we take into account the example of Figure 4. Matching description 1 against description 2, we obtain the following mappings:

$$\{(A1, A2), (E1, E2), (F1, F2), (G1, G2)\}$$

leading to the following subset of matching properties:

$$d = \{\#title, \#director, \#actor\}.$$

Since the example is very simple, it is easy to see how d is also the best candidate to denote an autonomous individual. In more complex situations, this approach can fail due to some properties in the candidate set which are in fact useless to individual discovery. However, the resulting candidate set can be used to perform matching again by considering only the properties in the candidate set, in order to verify the quality of the obtained results. A more detailed discussion on this is provided in the following section.

6 Experimental Results

In order to evaluate the proposed techniques, we take into account the case study presented in Section 2.1. In particular, we have two goals: i) we want to discover movies within Amazon.com DVD data; ii) we want to compare movie descriptions discovered in Amazon.com against movies in IMDB, in order to find different descriptions of the same real movies.

6.1 Discovery Test Case

Concerning the first goal, our approach for testing is to match all the DVD descriptions extracted from Amazon.com in order to find correspondences (i.e., mappings) among DVDs containing the same movie. The ground truth has been created by manually clustering DVDs with respect to the movie and by creating the set of expected mappings. It is important to stress that the goal here is not to find similar or equal DVDs, but DVDs, potentially different, with the same movie. This means that some data referred to the product, such as ISBN, is not useful to our needs. Moreover, data are quite "dirty" because they contain errors or incomplete information. Thus, we performed two preliminary operations: first, we execute matching by considering all the properties, in order to have a measure of how the discovery works without a set of properties that denote the autonomous individual movie (Test 1); second, we manually select the set of DVD properties referred to movies, and we execute matching again. This provides a measure of how the matching works in the ideal situation, when the relevant properties denoting movies in DVD descriptions are correctly selected (Test 3). Finally, we exploited the discovery function of HMatch to automatically retrieve the relevant properties for movies and we execute the matching by considering only the relevant properties (Test 2). In order to evaluate the quality of the results we used three measures: *precision*, defined as the quantity of retrieved mappings that are correct, and *recall*, defined as the quantity of correct mappings that are retrieved, and F-Measure which is the harmonic mean of the two. Results are shown in Table 2, where, for each Test, we provide two measures in the form X (Y), where X is the absolute value of precision or recall, while Y is the value of precision and recall with respect to the ideal matching case (Test 3).

The results show how matching over all the DVD properties is useless with respect to the goal of discovering movies. In fact, many DVD are similar because of properties like the format which is not relevant neither for the DVD identification nor for the movie identification. So, this method (Test 1) produces a high number of mappings, leading to a high recall with a very low precision.

Table 2. Experimental results for the discovery problem

Measure	Test 1	Test 2	Test 3
Precision	0.09 (0.13)	0.67 (0.83)	0.8 (1.0)
Recall	0.54 (0.96)	0.43 (0.76)	0.56 (1.0)
F-Measure	0.18 (0.26)	0.53 (0.79)	0.67 (1.0)

On the other way, manual selection of properties (Test 3) produces the best results that can be obtained by using matching for movie discovery. These results are characterized by a good precision and an average recall. This means that when we collect together DVDs with the same movie, it is quite probable that the collection is correct, even if some DVD is missing. The results obtained with the discovery function (Test 2) are promising because, despite the fact that the method is completely automatic, we obtained quite good precision and recall results, especially if compared with the ones obtained with the manual approach.

6.2 Matching Test Case

The matching test case has been executed by taking onto account data extracted from Amazon together with data provided by IMDB. The goal here is to evaluate in terms of precision and recall the quality of the results obtained using HMatch for individual comparison. In particular, we want to check if our instance matching techniques are suitable for automatically retrieve different descriptions denoting the same real entity (i.e., a movie). The ground truth has been defined by manually comparing the 27 movies retrieved in IMDB against the 87 DVDs retrieved on Amazon for the query "Scarface". In particular, we have manually mapped each movie with all the DVDs containing that movie. Then, we have executed HMatch by taking into account only the properties selected in the previous discovery test (Test 2). The results in terms of precision, recall and F-Measure are shown in Table 3.

Table 3. Experimental results for the matching problem

Precision	Recall	F-Measure
0.92	0.82	0.87

Results are in this case very good, which encourages us in our belief that instance matching techniques can be used as a support for the identification of entities over heterogeneous datasources.

7 Related Work

In the paper, we have seen how instance matching techniques are crucial for solving the identity recognition problem by exploiting automatic techniques.

In general, instance matching is frequently referred to as an *Entity Resolution* problem (also called *Deduplication* or *reference reconciliation*) and it is defined as *the process of identifying and merging records judged to represent the same real-world entity.* Up to now, the instance matching problem has been recognized as particularly relevant in database and data integration applications where it is referred to as *record linkage* and it is defined as *the task of quickly and accurately identifying records corresponding to the same entity from one or more data sources* [5].

A first important category of approaches to record linkage rely on statistical theories and techniques. In particular the problem is translated in a Bayesian inference problem [6] by means of decision rules based on probabilities, which are estimated using different techniques such as expectation maximization algorithms [7], with some further improvements in order to manage missing values and costs of misclassification. More recent approaches to record linkage are based on classification techniques in the machine learning. The supervised learning systems rely on the existence of training data in the form of record pairs, prelabeled as matching or not. When this kind of information is not available, other techniques must be adopted to detect matching records. In general these techniques define a distance metric for records which does not need tuning through training data. A simple approach is to consider a record as a unique string and to apply known edit-distance metrics [8] but, in such a way, references between different records are ignored. Ananthakrishna et al. [9] describe a similarity metric that uses not only the textual similarity, but the co-occurrence similarity of two entries in a database. A similar way to improve the comparison metrics is the one by Felix Naumann et al. on data fusion [10]. The similarity evaluation of two record is enhanced by the analysis of the context of a record, which is captured by considering not only the significant attributes of a record but also those attributes related to it through a foreign key. Naumann et al. exploit duplicate detection also to support schema matching functionalities. Halevy et al. [11] proposed an algorithm that, besides the exploitation of co-references among records, propagates information between reconciliation decisions of different descriptions to accumulate positive and negative evidences. Then, the algorithm gradually enriches references by merging attribute values. Our approach for instance matching has a similar strategy but we do not limit the context analysis to a single level of similarity propagation because the ontology instances generally have a more nested, tree-like, structure and the meaningful data is at the bottom of it. For this reason, ontology instance matching require more sophisticated techniques for handling the propagation of the similarity measure at the instance level. Moreover, for expressive languages (i.e. DLs) the structure of instances is not explicit and should be derived by reasoning.

In the Semantic Web the attention on instances for the purpose of ontology matching has been poorly studied and only basic techniques for ontology instance matching have been proposed. For example, in [12] instances are considered to support/validate concept matching techniques trough statistical analysis. This means that the similarity between two concepts is evaluated by measuring the "significance" in the overlap of their respective instance sets [13]. To this end, various similarity metrics have been proposed to evaluate instance similarity and thus instance-based concept matching [14]. Another important application of instance matching is ontology evolution, which means the need of supporting experts in managing ontology changes through advanced and possibly automated techniques. This is the case of the BOEMIE project (Bootstrapping Ontology Evolution with Multimedia Information Extraction) where a novel methodology for ontology evolution is defined to enhance traditional approaches and to provide methods and techniques for evolving a domain ontology through acquisition of semantic information from multimedia resources such as image, video, and audio [15].

8 Concluding Remarks

In this paper, we have discussed the problem of identity and entity recognition in the semantic web, by presenting three main sub-problems and by providing specific techniques to solve them. Our thesis is that instance matching and related techniques can be used to support entity recognition by enabling the automatic discovery of "hidden" entities as well as the automatic comparison of different descriptions of the same real-world objects. To this end we have presented promising experimental results. Concerning scalability, which is a crucial issue for semantic web applications, we are currently performing tests with larger datasets and our future work will be focused on optimization of the algorithms in terms of computational performances. Moreover we are collecting more experimental data in the BOEMIE European Project [5], where instance matching is used with the purpose of retrieving descriptions of the same entities over semantic data extracted from multimedia resources. BOEMIE provides methods and techniques for knowledge acquisition from multimedia content, by introducing the notion of evolving multimedia ontologies, which is used for the extraction of information from multimedia content in networked sources. In particular, multimedia resources are analyzed with the goal of providing a semantic description of their content. The result of this process is the population of an OWL ontology. In the process of population, different instances created from different resources are grouped together if they denote the same real-world object. Our current and future work in the project is to investigate, develop and test advanced instance matching techniques as a support for ontology population.

References

1. Bouquet, P., Stoermer, H., Mancioppi, M., Giacomuzzi, D.: OkkaM: Towards a Solution to the "Identity Crisis" on the Semantic Web. In: Proc. of the 3rd Italian Semantic Web Workshop (SWAP 2006), Pisa, Italy, CEUR (2006)
2. Castano, S., Ferrara, A., Montanelli, S.: Matching ontologies in open networked systems: Techniques and applications. Journal on Data Semantics (JoDS) **V** (2006)

[5] http://www.boemie.org

3. Bruno, S., Castano, S., Ferrara, A., Lorusso, D., Messa, G., Montanelli, S.: Ontology Coordination Tools: Version 2. Technical Report D4.7, BOEMIE Project, FP6-027538, 6th EU Framework Programme (2007)
4. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer-Verlag (2007)
5. Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S.: Duplicate record detection: A survey. IEEE Transactions on Knowledge and Data Engineering **19** (2007)
6. Fellegi, I.P., Sunter, A.B.: A theory for record linkage. Journal of the American Statistical Association **64** (1969)
7. Winkler, W.: Improved decision rules in the fellegi-sunter model of record linkage. In: Proceedings of the Section on Survey Research Methods. (1993)
8. Monge, A.E., Elkan, C.: An efficient domain-independent algorithm for detecting approximately duplicate database records. In: Research Issues on Data Mining and Knowledge Discovery. (1997)
9. Ananthakrishna, R., Chaudhuri, S., Ganti, V.: Eliminating fuzzy duplicates in data warehouses. In: In Proceedings of the 28th International Conference on Very Large Databases (VLDB 2002). (2002)
10. Naumann, F., Bilke, A., Bleiholder, J., Weis, M.: Data fusion in three steps: Resolving schema, tuple, and value inconsistencies. IEEE Data Eng. Bull. **29** (2006) 21–31
11. Dong, X., Halevy, A., Madhavan, J.: Reference Reconciliation in Complex Information Spaces. In: Proc. of the ACM SIGMOD Int. Conference on Management of Data, Baltimore, Maryland, USA (2005)
12. Wang, C., Lu, J., Zhang, G.: Integration of Ontology Data through Learning Instance Matching. In: Proc. of the IEEE Int. Conference on Web Intelligence (WI'06), Hong Kong, China (2006)
13. Isaac, A., van der Meij, L., Schlobach, S., Wang, S.: An Empirical Study of Instance-Based Ontology Matching. In: Proc. of the 6th Int. Semantic Web Conference, 2nd Asian Semantic Web Conference (ISWC 2007+ASWC 2007), Busan, Korea (2007)
14. Engmann, D., Massmann, S.: Instance Matching with COMA++. In: Proc. of the Workshop on Datenbanksysteme in Business, Technologie und Web (BTW 2007), Aachen, Germany (2007)
15. Castano, S., Espinosa, S., Ferrara, A., Karkaletsis, V., Kaya, A., Melzer, S., Möller, R., Montanelli, S., Petasis, G.: Multimedia Interpretation for Dynamic Ontology Evolution. Journal of Logic and Computation, special issue on Ontology Dynamics (2008) To appear.