Instance Matching for Ontology Population [⋆]
Paper type: FULL PAPER

Silvana Castano, Alfio Ferrara, Stefano Montanelli, Davide Lorusso

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

Abstract. In the context of ontology evolution, ontology population is the activity of acquiring new semantic descriptions of data extracted from heterogeneous data sources. To this end, the capability of comparing several instances extracted from different sources is crucial. In this paper, we focus on the problem of *instance matching* and its role for ontology population. Moreover, we present the instance matching systems HMatch($\mathcal{I}$) which has been developed in the framework of the BOEMIE research project.
Keywords: ontology population, instance matching

1 Introduction

In the Semantic Web, ontology evolution is getting more and more importance due to the need of supporting experts in managing ontology changes through advanced and possibly automated techniques [1]. This is the goal of the BOEMIE project[1] (Bootstrapping Ontology Evolution with Multimedia Information Extraction) where a novel methodology for ontology evolution is defined to enhance traditional approaches by providing methods and techniques for the acquisition of new knowledge from a wide variety of multimedia resources [2]. In this paper, we focus on the problem of *instance matching* and its role for ontology population in BOEMIE. For ontology population, the availability of instance matching techniques plays a crucial role to correctly perform the insertion activity and to discover the relationship between the new incoming instance and the set of instances already stored in the ontology. In this respect, we describe the HMatch($\mathcal{I}$) component of the HMatch 2.0 ontology matching suite [3]. This component has been specifically developed in the framework of the BOEMIE project to support population activities by providing instance-level matching techniques devoted to evaluate the similarity of different ontology instances expressed in form of OWL ABoxes. In particular, the instance matching algorithm of HMatch($\mathcal{I}$) will be presented together with experimental results obtained on a real case-study.

The paper is organized as follows. In Section 2, we discuss the applicability of instance matching to three main contexts by focusing in particular on ontology

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

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

population. In Section 3, we present the approach followed for instance matching in HMatch($\mathcal{I}$). In Section 4, a more detailed view of the matching procedure of HMatch($\mathcal{I}$) is given, together with an example. Experimental results on BOEMIE data are discussed in Section 5. In Section 6, related work on instance matching is presented. In Section 7, we provide our concluding remarks.

2 Roles of instance matching for ontology population

Techniques for ontology instance matching are required in several application contexts where the capability of comparing different individuals with the goal of recognizing the same real-world object is demanded. In particular, the application of instance matching is crucial in the following contexts.

Semantic integration. Due to the increasing popularity of Semantic Web technologies, a novel attention on semantic integration issues has raised. In this respect, most of the recent research effort has been focused on ontology matching and alignment with the aim to enforce advanced techniques for (semi) automatically discovering semantic mappings between possibly distributed and heterogenous ontologies. For semantic integration, advanced techniques for ontology instance matching are required to correctly combine data describing individuals in different sources and to improve the accuracy of the ontology alignment process.

Identity recognition. Identity recognition is an emerging topic in the Semantic Web field and it refers to the capability of detecting whether two different resource descriptions refer to the same real-world entity, namely an individual. As discussed in [4], in developing any kind of semantic-driven information system possibly based on knowledge exchange and reuse, the phases of ontology conceptualization and population need to be clearly distinguished. Such a distinction is motivated by the observation that the same set of individuals can be used to populate different ontologies also in different domains.

Ontology population. In modern Semantic Web applications, ontology evolution is becoming more and more important due to the need of supporting experts in managing ontology changes through advanced, and possibly automated, techniques. One of the main activities in ontology evolution is ontology population, where the ontology is evolved by acquiring new semantic descriptions of data extracted from heterogeneous data sources. For ontology population, instance matching plays a crucial role to correctly perform the insertion activity and to discover the relationship between new incoming instance and the set of instances already stored in the ontology. In this respect, instance matching has the role of providing a set of semantic mappings between incoming instances and those already stored. The mappings produced by instance matching are exploited to cluster those instances that are recognized as referring to the same real-world entity.

In this paper, we will focus on the problem of ontology population, by discussing the approach adopted in the BOEMIE project and the role of instance matching in this framework.

2.1 Ontology population in BOEMIE

In BOEMIE, ontology population is defined as the process of inserting new instances in a given domain ontology and it is seen as a particular case of ontology evolution. In particular, focus of the BOEMIE project is *multimedia ontology evolution* with the goal of providing advanced methods and techniques for evolving a domain ontology through acquisition of semantic information from multimedia sources such as image, video, and audio. A novel methodology for ontology evolution is defined to this end, which is characterized by a pattern-driven approach where the evolution activities are determined according to the results of a semantic interpretation process performed over the information extracted from the underlying multimedia sources [2]. More in detail, ontology evolution in BOEMIE is defined as a multi-step process composed of the following tasks:

- *Multimodal information extraction* where a multimedia document is analyzed to provide an ABox representation of the real-world objects contained in the document.
- *Semantic interpretation* where the ABoxes produced by extraction are processed with standard (i.e., deduction) and non-standard (i.e., abduction) reasoning techniques in order to determine how they can be associated with the concepts in the ontology.
- *Pattern selection and evolution* where the most appropriate evolution pattern is selected on the basis of the semantic interpretation results and where the corresponding evolution actions on the ontology are triggered accordingly.

In this respect, in BOEMIE, ontology population is defined as an evolution action and it is invoked when a *single explanation pattern* is detected by the semantic interpretation. This means that population regards the insertion in the domain ontology of the extracted ABoxes when they are recognized as instances of a concept already stored in the ontology.

2.2 The roles of instance matching

In BOEMIE, the ontology population process is supported, for what concerns instance matching, by using the instance matching tool $\mathsf{HMatch}(\mathcal{I})$, developed as a component of the HMatch 2.0 matching system [3]. The role of instance matching in BOEMIE is twofold.

- To identify different instances denoting the same real-worl object. This is achieved by comparing the different Abox descriptions of instances in order to evaluate their similarity with respect to those assertions that are more relevant for object identification. Comparison is calculated under the assumption that the similarity denotes a degree of identity. Goal of this activity is to support the correct assimilation of an instance, which is achieved by clustering all similar instances and by defining a new individual representative of the whole cluster.

- To detect instances that denote similar individuals. In this case, similarity is not seen as a degree of identity, but just as a measure of the properties that two objects share. This is achieved by comparing assertions without distinguishing among those relevant for object identification and others not relevant to this end. Goal of this activity is to support proximity-base search of data. For example, taking into account the athletics domain in BOEMIE, this mappings are used in order to find similar athletes or sport events, given a object of interest.

3 Instance matching with HMatch($\mathcal{I}$)

HMatch($\mathcal{I}$) has been developed with the goal of addressing the main requirements for instance matching in general and for ontology population in particular. A first requirement is the capability of comparing structurally heterogeneous instance representations, in particular when population is performed with respect to more than one Tbox. To this end, HMatch($\mathcal{I}$) supports the possibility of being combined with traditional concept-level matching techniques, provided either by HMatch 2.0 or by other state-of-the-art matching systems [5]. A second requirement is the capability of considering up to the complete set of instance assertions. It refers to the capability of HMatch($\mathcal{I}$) to differently compose the set of considered assertions for a given instance. Two different approaches are supported. An *syntactic* approach can be followed where the set of instance assertions considered for matching are determined by uniquely relying on the explicitly defined knowledge of an ontology. This is suited when computation time is a critical factor and/or the interface with the reasoning service is not available. A *semantic* approach can be followed where the set of explicit instance assertions is extended with the set of implicit assertions obtained through an inference mechanism, by materializing aboxes that need to be compared. This is suited when accuracy and completeness of the HMatch($\mathcal{I}$) results are more relevant than performance. A third requirement is the capability of capturing the identification power of instance properties. It refers to the capability of HMatch($\mathcal{I}$) to distinguish between featuring properties, namely those properties with high identification power, and non-featuring properties which poorly contribute to individual identification. Clearly, the capability to *automatically* detect featuring properties is preferable, although not feasible in all matching cases. For example, HMatch($\mathcal{I}$) can automatically learn information about featuring properties by applying statistical techniques based on the observation of recurring patter values in a set of homologous instances. Intuitively, the idea is that properties with distinct values for each considered instance could have a high identification power. However, the availability of a training set of instances can not be guaranteed. In this case, HMatch($\mathcal{I}$) offers an interactive way of manually setting a weight to assess the identification power of instance properties.

A detailed description of the instance matching process of HMatch($\mathcal{I}$) is shown in Figure 1. The process starts with the acquisition of the instances to be compared from the knowledge base. Depending on the HMatch($\mathcal{I}$) configuration, we exploit HMatch 2.0 and a reasoning service in order to produce concept mappings and a materialized version of the Abox(es), respectively. If instances refer to different Tboxes, TBox mappings are exploited to run property values and instance comparisons. In fact, only data and instances that are members of matching properties and classes are compared. The materialized version of the Abox(es) is used to consider all the property values featuring an instance, both the ones explicitly asserted and the ones derived by reasoning. The subsequent step is to define a weight for property assertions (*featuring properties identification*). Goal of this step is to give more relevance to those assertions that are considered as more important for individuals identification during the matching process. The subsequent matching stage is based on the comparison of instance properties, also called *roles*, and corresponding property values, also called *role fillers*. To this end, each instance in HMatch($\mathcal{I}$) is represented as a tree (*instance tree construction*) where role fillers are nodes and roles are labeled edges. Matching is then performed by traversing in postorder the instance trees of the two instances and by recursively executing filler similarity or instance similarity on corresponding nodes according to their type. When nodes are datatype values, their similarity is computed by relying on specific matching functions (datatype matchers) that vary according to the considered datatype (*filler similarity evaluation*). When nodes are instances, the similarity is evaluated by considering

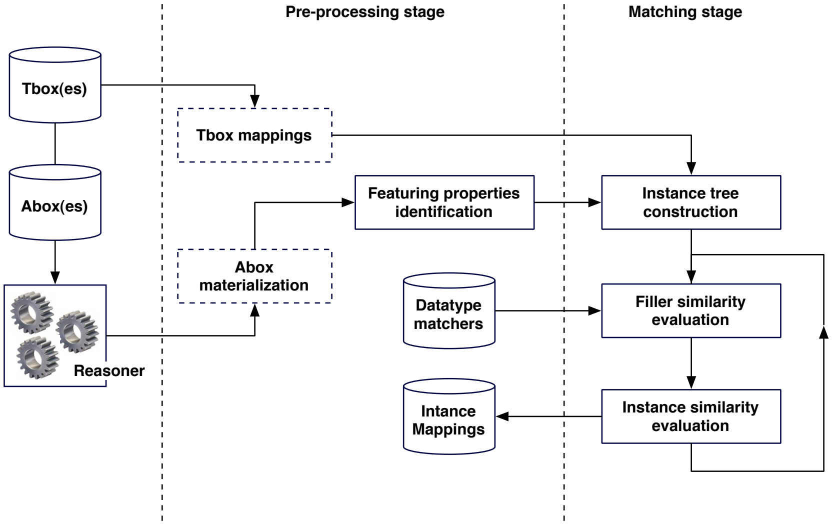

Fig. 1. The instance matching process of HMatch($\mathcal{I}$)

the matching value of all the nodes in their respective sub-trees previously traversed (*instance similarity evaluation*). The set of resulting instance mappings are finally returned.

4 The HMatch($\mathcal{I}$) matching procedure

The HMatch($\mathcal{I}$) matching procedure is based on two main functions, namely *Instance affinity* (IA) and *Filler similarity* (F_S).

Instance Affinity. Given two individuals i_1 and i_2 that are instances of the same (or matching) concept, the instance affinity function $IA(i_1, i_2) \rightarrow [0, 1]$ provides a measure of their affinity in the range [0,1]. For each individual i, instance affinity is calculated by taking into account all the properties $p_1^i, \ldots, p_n^i$ featuring i together with their corresponding property fillers $f_1^i, \ldots, f_n^i$. Each property p_j, is associated with a weight $W_{p_j} \in [0, 1]$ expressing the capability of p_i for the goal of univocally identifying the individual i in the domain of interest. This weight is defined during the *featuring properties identification* step of the instance matching process. In BOEMIE, property weights are manually defined for the considered domain by taking into account also the results of the extraction process from a corpus of (manually) annotated multimedia resources.

Definition 1. *Instance affinity. Given two individuals i_1 and i_2, the instance affinity function $IA(i_1, i_2)$ between them is calculated as follows:*

$$IA(i_1, i_2) = \frac{2 \cdot \sum_{k=1}^{k=n} F_S(f_k^{i_1}, f_k^{i_2}) \cdot W_{p_k^{i_1}}}{\sum_{k=1}^{k=n} W_{p_k^{i_1}} + \sum_{k=1}^{k=n} W_{p_k^{i_2}}}$$

For each property filler $f_k^{i_1}$ featuring i_1, we execute the function $F_S(f_k^{i_1}, f_k^{i_2}) \rightarrow \{0, 1\}$ between $f_k^{i_1}$ and the corresponding role filler $f_k^{i_2}$ of i_2, where two fillers corresponds if they denote the value of the same (or matching) property. The filler similarity function returns 1 if the two fillers match, and 0 otherwise. The goal of this step is to consider only the matching property fillers of the two individuals. Then, we sum the weights $W_{p_k^{i_1}}$ associated with each property filler. Note that we take into account only the weights of the properties featuring i_1, since the same (or matching) properties are featured with the same weight. Finally, we calculate the ratio of the sum of weights associated with matching filler properties and the sum of the weight of all the properties featuring the two individuals. During the comparison of two individuals there can be situations in which a defined filler for a property is missing, which means that we have no information about that property at all. In this case, we can use two different strategies: pessimistic or optimistic. The former considers the lack of information as an evidence of the difference between two individuals; the latter ignores the missing property value, interpreting it as undefined, so it does not take part in the computation of similarity. In HMatch($\mathcal{I}$), the optimistic strategy is adopted because we want to evaluate the knowledge explicitly asserted about an individual. Then if all the essential properties, that is the ones which differentiate each individual from the others, are valued, the absence of the other properties has not influence on the similarity evaluation. Property fillers can be datatypes (e.g., strings, numbers, dates) or other individuals. In order to adopt the matching technique more suitable for a given pair of fillers, we define the specific function filler similarity.

Filler Similarity. The filler similarity function $F_S(f_1, f_2) \rightarrow \{0, 1\}$ previously introduced is defined as follows:

Definition 2. *Filler similarity. Given two property fillers f_1 and f_2 and a threshold t, the filler similarity function $F_S(f_1, f_2) \rightarrow \{0, 1\}$ is defined as:*

$$F_S(f_1, f_2) = \begin{cases} 1, \textit{ if } sim(f_1, f_2) \geq t; \\ 0, \textit{ otherwise} \end{cases}$$

$Sim(f_1, f_2)$ is a value in the range [0,1] and is calculated in different ways depending on the type of the role fillers f_1 and f_2, according to the following rules:

- **Rule 1:** if f_1 and f_2 are both datatypes, a suitable matcher is executed which evaluates the similarity between datatype values according to the semantic meaning of the properties and to their datatype category. A detailed description of these datatype matchers is provided in [6].
- **Rule 2:** if f_1 and f_2 are both individuals, we check if they are featured by property assertions. If not, $sim(f_1, f_2)$ returns 1 if f_1 and f_2 are instances of the same (or matching) concepts, and 0 otherwise. If f_1 and f_2 are featured by property assertions, $sim(f_1, f_2) = IA(f_1, f_2)$, thus leading to a recursive step, which is iterated until all the datatype property fillers are compared.
- **Rule 3:** if f_1 is a dataype and f_2 is an individual (or viceversa), we check is f_2 is featured by a property filler f_k matching with f_1. In this case, we apply Rule 1 in order to obtain $sim(f_1, f_2) = sim(f_1, f_k)$. Otherwise, $sim(f_1, f_2) = 0$.
- **Rule 4:** if f_1 and/or f_2 are a collection of values, such as in case of properties with multiple values, $sim(f_1, f_2)$ is calculated by evaluating a set similarity measure over the two collections of values.

Example. Consider two individuals *PersonName_7368* and *PersonName_4352* extracted from the analysis of a web page in the athletics domain of BOEMIE. The two individuals are instances of the concept *PersonName* and their properties as shown in Figure 2.

The two individuals denote (different) person names referring to different athletes. The two considered individuals share a high number of common characteristics. In fact, the two athletes come from the same country (i.e., Poland), have the same gender (i.e., male), and are associated with the same performance (i.e., 2.36). So, the instance matching should be able to recognize the athlete

(PersonName_7368, ‘Michal BIENIEK’) : hasPersonNameValue
(PersonName_7368, Country_3415) : personNameToCountryName
(PersonName_7368, Performance_4389) : personNameToPerformance
(PersonName_7368, Male_640) : personNameToGender
(Country_3415, ‘POL’) : hasCountryNameValue
(Performance_4389, 2.36) : hasPerformanceValue

(a) ABox of individual PersonName_7368

(PersonName_4352, ‘Artur PARTYKA’) : hasPersonNameValue
(PersonName_4352, Country_5567) : personNameToCountryName
(PersonName_4352, Performance_6732) : personNameToPerformance
(PersonName_4352, Male_640) : personNameToGender
(Country_5567, ‘POL’) : hasCountryNameValue
(Performance_6732, 2.36) : hasPerformanceValue

(b) ABox of individual PersonName_4352

Fig. 2. Two individuals of the concept *PersonName* in the BOEMIE sport ontology

similarity while capturing the identity diversity of the two athletes at the same time.

Since the similarity between *PersonName_7368* and *PersonName_4352* depends on the similarity among their property fillers, the evaluation of such a similarity is based on the results obtained by applying the filler similarity function, that are:

$$F_S(Country\_3415, Country\_5567) = 1$$

$$F_S(Performance\_4389, Performance\_6732) = 1$$

$$F_S(Male\_640, Male\_640) = 1$$

$$F_S(`MichalBIENIEK', `ArturPARTYKA') = 0$$

In order to deal with the fact that the two individuals are referred to different persons in spite of the fact that they share a high number of features, we need to capture the intuition that some properties, such as the name, are more important than others, such as the performance, for the sake of object identification. In the example, we rely on property identification weights defined for the athletic domain of BOEMIE, i.e., $W_N = 1.0$ for *hasPersonNameValue*, $W_G = 0.3$ for *personNameToGender*, $W_C = 0.3$ for *personNameToCountryName*, and $W_P = 0.0$ for *personNameToPerformance*, respectively. These weights state that the name is considered to be the most important attribute for the identification of a person. Gender is relevant, since it does not change in time, but it is not a key of a person record because there are many person with the same gender. The same can be said about the country. Finally, the performance is not relevant for the identification of a person because it is not an attribute of the person but a result obtained by the person in some kind of sport event. On the basis of these weights, we apply the instance affinity function described above, with the following results:

$$IA(P_{7368}, P_{4352}) = \frac{W_G + W_C + W_P}{W_P + W_G + W_C + W_P} = \frac{0.3 + 0.3 + 0.0}{1.0 + 0.3 + 0.3 + 0.0} = 0.375$$

The example shows how, using the instance affinity function and the property weights, $\mathsf{HMatch}(\mathcal{I})$ is capable of providing a measure of identification among the two individuals. We note that, if we apply the same weight to all the properties, the instance affinity function can be used also to evaluate a general degree of similarity between two individuals rather than the fact that they denote the same real object in the domain. For the goals of BOEMIE in the instance grouping task of the population activity the aim is to group together individuals which represent the same real object. Thus, specific weights for the properties in the BOEMIE ontology are used.

5 Experimental results

The methodological approach commonly used for the evaluation of the semantic matchmaking tools is based on the idea of building a benchmark constituted by several heterogeneous ontologies to be matched and a set of manually defined results, that is a set of expected mappings (E_M). Then, the matching tool to be evaluated is executed against the ontologies in the benchmark, in order to obtain a set of automatically retrieved mappings (R_M). On the basis of E_M and R_M, the following metrics are then defined for the evaluation of the tool.

Definition 3. *Precision. Precision is the fraction of the mappings retrieved that are relevant with respect to the benchmark. More formally, given the set of retrieved mappings R_M and the set of expected mappings E_M, precision P is defined as:*

$$P = \frac{\mid E_M \cap R_M \mid}{\mid R_M \mid}$$

Definition 4. *Recall. Recall is the fraction of the mappings that are relevant in the benchmark and that are successfully retrieved by the tool. More formally, given the set of retrieved mappings R_M and the set of expected mappings E_M, recall R is defined as:*

$$R = \frac{\mid E_M \cap R_M \mid}{\mid E_M \mid}$$

Concerning the evaluation of instance matching, we have created a specific benchmark by taking into account 26 BOEMIE ABoxes extracted by textual modality from textual resources about High Jump events. The benchmark involves 15841 instances. Among these instances, we have focused on instances from the concept *PersonName* and from the concept *Athlete*, and we have manually defined a set of 388 mappings, which represent the instance correspondences that are expected. By using this set of expected mappings, we then evaluate instance matching by calculating precision and recall.

For the test, since we were interested in finding athletes records, we have set an identification weight of 1.0 for person names and of 0.3 for nationalities and ages. Then, we have selected a group of athletes, namely ‘Michal BENIEK’, ‘Fabricio ROMERO’, ‘Ebba JUNGMARK’, and ‘Germaine MASON’. The benchmark contains, for these athletes, the number of instances shown in Table 1. it is easy to see that, for each concept, given the number n of instances expected for each athlete, the number of m of mappings expected for each athlete is $m = \frac{n(n-1)}{2}$, leading to Table 2. Thus, for the whole benchmark the number of expected mappings is 388. We have executed $\mathsf{HMatch}(\mathcal{I})$ against the benchmark and, then, we have produced the transitive closure of the set of mappings retrieved by $\mathsf{HMatch}(\mathcal{I})$, in order to capture also the mappings produced by the transitive interpretation of similarity. The results are shown in Table 3.

Table 1. Number of expected instances per person in the benchmark

Concept	**BENIEK**	**ROMERO**	**JUNGMARK**	**MASON**
PersonName	7	3	3	22
Athlete	7	3	2	15

Table 2. Number of expected mappings per person in the benchmark

Concept	**BENIEK**	**ROMERO**	**JUNGMARK**	**MASON**
PersonName	21	3	3	231
Athlete	21	3	1	105

Looking at the results, we can conclude that $\mathsf{HMatch}(\mathcal{I})$ is featured by a very high precision and a low recall. This means that the results retrieved by $\mathsf{HMatch}(\mathcal{I})$ are highly reliable, but we do not capture all the expected results. Thus, the current version of $\mathsf{HMatch}(\mathcal{I})$ can be really useful for suggesting similar instances by taking into account a specific instance of interest, since the results obtained are precise. For the next version of $\mathsf{HMatch}(\mathcal{I})$ we will work on the goal of increasing the recall. The transitive closure shows that a combination of the results obtained from a single execution of $\mathsf{HMatch}(\mathcal{I})$ can be useful to this goal. Another promising solution is based on the idea of modifying the parameters and improve the automatic mechanism for detecting identification weights for properties.

Table 3. Results of instance matching evaluation

	HMatch($\mathcal{I}$)	HMatch($\mathcal{I}$) **with transitive closure**
E	388	388
R	165	226
$R \cap E$	159	202
Precision	0.96	0.89
Recall	0.41	0.58

Legenda: E = expected mappings; R = retrieved mappings

6 Related work

In the paper, we have seen how instance matching techniques are crucial for supporting ontology population. In the literature, instance matching has been widely investigated in several application domains where it is known with different names according to the specific requirements that need to be satisfied and to the goals that need to be pursued. 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* [7]. More in general, instance matching is frequently referred to as an *Entity Resolution* problem (also called *Deduplication* [8] or *reference reconciliation* [9]) and it is defined as *the process of identifying and merging records judged to represent the same real-world entity* [10]. In the Semantic Web, a novel attention on semantic integration issues has raised. In this respect, most of the recent research effort has been focused on ontology matching and alignment with the aim to enforce advanced techniques for (semi) automatically discovering semantic mappings between possibly distributed and heterogenous ontologies. Up to now, the attention on instances for the purpose of ontology matching has been poor and only basic techniques for ontology instance matching have been proposed. For example, in [11] instances are considered to support/validate concept matching techniques trough statistical analysis. This means that the similarity between two concept is evaluated by measuring the "significance" in the overlap of their respective instance sets [12]. To this end, various similarity metrics have been proposed to evaluate instance similarity and thus instance-based concept matching [13].

7 Concluding remarks and future work

In this paper, we have discussed instance matching and its applicability to the problem of ontology population. In particular, we have presented the population approach proposed in the BOEMIE project. Ongoing and future work are addressing the following issues. First, refinement and experimentation of

$\mathsf{HMatch}(\mathcal{I})$ with more specific data matchers and by using different matching thresholds for ontology population. In population, the algorithm precision is crucial, more than recall. Thus, we will work on the $\mathsf{HMatch}(\mathcal{I})$ configuration in order to set up a population-specific matching policy. Second, the extension of $\mathsf{HMatch}(\mathcal{I})$ for general-purpose instance matching. To this and, we work on a better integration of $\mathsf{HMatch}(\mathcal{I})$ with schema and concept matchmaking tools with the goal of using schema mappings to perform instance matching between highly heterogeneous ontologies. Moreover, we work on recall in order to increase this value in controlled experiments.

References

1. Flouris, G.: On Belief Change in Ontology Evolution. AI Communications **19** (2006)
2. Castano, S., Espinosa, S., Ferrara, A., Karkaletsis, V., Kaya, A., Melzer, S., Moller, R., Montanelli, S., Petasis, G.: Multimedia Interpretation for Dynamic Ontology Evolution. Journal of Logic and Computation, special issue on Ontology Dynamics (2008) To appear.
3. Castano, S., Ferrara, A., Montanelli, S.: Matching Ontologies in Open Networked Systems: Techniques and Applications. Journal on Data Semantics (JoDS) **V** (2006)
4. 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, Pisa, Italy (2006)
5. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer-Verlag (2007)
6. 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)
7. Gu, L., Baxter, R., Vickers, D., Rainsford, C.: Record Linkage: Current Practice and Future Directions. Technical Report 03/83, CSIRO Mathematical and Information Sciences, Canberra, Australia (2003)
8. Sarawagi, S., Bhamidipaty, A.: Interactive Deduplication using Active Learning. In: Proc. of the 8th ACM SIGKDD Int. Conference on Knowledge Discovery and Data Mining, Alberta, Canada (2002)
9. 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)
10. Menestrina, D., Benjelloun, O., Garcia-Molina, H.: Generic Entity Resolution with Data Confidences. In: Proc. of the 1st Int VLDB Workshop on Clean Databases (CleanDB 2006), Seoul, Korea (2006)
11. 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)
12. 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 (ISWC 2007), Busan, Korea (2007)
13. 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)