Dealing with Matching Variability of Semantic Web Data Using Contexts

Silvana Castano, Alfio Ferrara, and Stefano Montanelli

Università degli Studi di Milano,
DICo, via Comelico 39, 20135 Milano, Italy
{castano,ferrara,montanelli}@dico.unimi.it

Abstract. Goal of this paper is to propose a reference modeling framework to explicitly identify and formalize the different levels of variability that can arise along all the involved dimensions of a matching execution. The proposed framework is based on the notion of *knowledge chunk*, *context*, and *mapping* to abstract the variability levels and related operations along the *source-dataset*, the *matching-dataset*, and the *mapping-set* dimensions, respectively. An application of the proposed framework with instantiation in the HMatch 2.0 systems is illustrated.

Keywords: Variability modeling, matching, semantic web, knowledge management.

1 Introduction

Techniques for matching semantic web data are an essential component of modern information management and evolution architectures, to correctly compare and integrate disparate resource descriptions and to promote effective resource sharing and integration on the global scale [1]. Several tools and approaches have been proposed in the literature for performing ontology and schema matching [2,3], and, more recently, also for performing instance matching [4,5]. A key demand for effective matching tools and techniques is the capability to deal with different kinds of variability that can emerge during a matching execution on a certain dataset. For example, one must be able to deal with variability of data representations both from a linguistic and a structural point of view, with the purpose of dynamically isolating only the subset of data relevant for a certain matching goal. As another example, the output of the matching process can result in mappings at different levels of granularity, providing just a similarity measure expressing that pairs of concepts or instances match as a whole, or more detailed information about mapping rules to convert their matching properties one to another. For effective matching of semantic web data, a systematic analysis and modeling of the various kinds of variability that influence a matching execution are required, and a conceptual framework for their classification is still missing in the field.

Goal of this paper is to propose a reference modeling framework to explicitly classify and formalize the different levels of variability that can arise along all

B. Pernici (Ed.): CAiSE 2010, LNCS 6051, pp. 194–208, 2010.

the involved dimensions of a matching execution. The proposed framework is based on the notion of *knowledge chunk*, *context*, and *mapping* to abstract and formalize the variability levels and related operations along the *source-dataset*, the *matching-dataset*, and the *mapping-set* dimensions, respectively. An application of this framework to show its instantiation in the HMatch 2.0 system is also illustrated. We want to stress that we do not propose yet another matching approach, rather we want to focus on the problem of modeling the matching variability per se'. To the best of our knowledge, this is a novel contribution in the field of matching, with can be taken as a reference for i) providing a disciplined guidance for the design of new matching tools/techniques, flexibly customizable to operate in different situations at both schema and instance level and ii) defining a conceptual framework to analyze and compare existing systems/approaches with respect to the degree of variability actually supported.

The paper is organized as follows. After discussing related work in Section 2, in Section 3, we introduce the basic notions of the proposed Matching Variability Framework. Details about the formalization of contexts for modeling matching variability along each dimension are presented in Section 4, 5, and 6. In Section 7, we discuss an example of instantiation in the HMatch 2.0 system. Finally, concluding remarks are provided in Section 8.

2 Related Work

Relevant work on matching variability exists in the literature on ontology and schema matching [2,3,6]. However, we observe that in most of the existing approaches, variability is only partially supported or it is considered but in an implicit way by the various tools and prototypes. In particular, variabilities at the matching-dataset level are "embedded" in the tools through some level of configurability of the matching process. In this direction, a relevant example is provided by tools like ASMOV [7], RiMOM [8], and DSSim [9], as well as in mostly theoretical approaches like [10]. More recent work in the field of ontology matching is mainly focused on investigating the variabilities at the level of the mapping-set. In particular, recent approaches are being proposed to discuss how the mappings produced by different tools can be combined on the basis of different similarity measures. In [11], the variability of the possible mapping combinations is exploited to improve the mapping quality in terms of precision and recall, rather than to work on the modeling aspects of mapping combination. In this respect, interesting work are provided in [12,13], where the focus is more on presenting the possible operations that can be performed over a mapping-set, rather than on discussing the variabilities of the matching execution that originates them. Two other kinds of approaches to the problem of mapping exploitation are given in the framework of query answering [14] and of reasoning-based mapping validation [15,16]. In both the cases, however, the proposed techniques do not take into account the problem of modeling different mapping exploitation/validation strategies in correspondence of different matching contexts and/or goals. We note that all the presented tools/approaches are mainly focused on discussing

the aspects of variability at the level of both matching-dataset and mapping-set, while matching variabilities at the level of the source-dataset have not been formalized yet. With respect to the related work, this paper is a first attempt to give a comprehensive formalization of matching variability, by considering all the different dimensions involved in a matching execution.

3 Modeling Matching Variability

Data and ontology matching is frequently invoked in different scenarios and with different purposes. For this reason, the requirements and conditions under which matching is performed can vary from one case to another according to the specific goal that needs to be satisfied. This means that different types and different levels of variability characterize each matching execution. We represent these levels of variability along the following three dimensions: *variability of the source-dataset*, *variability of the matching-dataset*, and *variability of the mapping-set*.

Variability of the source-dataset. This kind of variability describes the different customizations that can be performed over the initial (i.e., source) dataset considered for matching. In particular, this variability dimension expresses the possible filtering operations that can be applied to the source-dataset to restrict and to better focus on the data of interest for the considered matching execution. We distinguish two different levels of variability along this dimension:

- *Abstract selection (α).* It allows to filter out the source-dataset by selecting for matching only a subset of the data properties according to a given criterion.
- *Concrete selection (χ).* It allows to filter out the source-dataset by selecting for matching only those data that exhibit a certain value for a given (set of) property.

As a result of the application of these levels of variability to the source-dataset, the so-called matching-dataset is produced.

Variability of the matching-dataset. This kind of variability describes the different customizations on the accuracy/deepness of the matching execution that can be performed over the matching-dataset. We distinguish three different levels of variability along this dimension:

- *Constraint matching (κ).* It allows to specify a precondition (i.e., a constraint) that has to be verified by the data in the matching-dataset for being matched. This way, it is possible to subordinate the matching execution to those data that satisfy the given constraint.
- *Scope matching (π).* It allows to specify at which level of deepness the matching execution has to be performed in terms of number and kind of features to consider for the data comparison.

– *Weight matching (ω).* It allows to assign a different level of relevance to the properties of the data to be matched. This way, it is possible to specify that one or more properties have a higher relevance than others within a given matching execution.

Variability of the mapping-set. This kind of variability captures the different customizations on the mappings that are returned as result of a matching execution. We distinguish three different levels of variability along this dimension:

– *Mapping ranking (ρ).* It allows to specify a minimum threshold that should be satisfied by a mapping to consider its corresponding elements as matching elements.
– *Mapping cardinality (δ).* It allows to specify the number of correspondences that are admitted in the matching result for a given element e. The choice spans from the one-to-one option, where only the mapping with the best matching element of e is included in the result, up to the many-to-many option, where all the discovered mappings with the matching elements of e are included in the result.
– *Mapping granularity (γ).* It allows to specify the level of granularity of the mappings produced as a result of the matching execution. The choice spans from a generic element-to-element correspondence, up to a complete mapping table of correspondences between the single properties of two matching elements.

The levels of variability along the three dimensions described above can be differently combined to obtain a specific configuration of the matching execution. All the possible matching variabilities are described by the three-dimension schema of Figure 1(a), where the nature of a matching execution is determined by the variability levels activated (+) or not activated (-) along the three dimensions. A given combination of choices depends on the specific target to satisfy. As an example in a bibliographic scenario, we consider the retrieval of publications of the same authors in the years from 2000 to 2003. This matching target can be satisfied by the combination $\langle +\alpha, +\chi, -\kappa, +\pi, -\omega, +\rho, +\delta, -\gamma \rangle$ (Figure 1(b)). The abstract selection ($+\alpha$) allows to restrict the matching execution to the comparison of authors, while the concrete selection ($+\chi$) allows to consider only publications from 2000 to 2003, respectively. The scope matching ($+\pi$) allows to set the maximum level of deepness for the comparison of authors of two publications by considering all the information available in the corresponding semantic web representations. Finally, the mapping ranking ($+\rho$) and the mapping cardinality ($+\delta$) allow to specify a matching threshold t and a many-to-many option for mapping cardinality, respectively. This way, for a given publication, all the publications with a matching value over t will be returned in the matching result. As another example, the matching configuration for discovering all the bibliographic records referring to the same real publication is shown in Figure 1(c) and it will be discussed in Section 7.

3.1 The Matching Variability Framework

To model matching variabilities modeling, we introduce the *Matching Variability Framework*, based on the notions of *knowledge chunk*, *mapping*, and *context*.

Knowledge chunk. A knowledge chunk kc represents an element of interest for matching, either concept or instance. Given a semantic web resource $\mathcal{O}$, like a RDF(S) repository or an OWL ontology, let $\mathcal{N}$ be the set of element names in the signature of $\mathcal{O}$, $\mathcal{P}$ is a set of property/relation names, $\mathcal{L}$ the set of datatypes and literal values in $\mathcal{O}$, and id the identifier of O, such as its URI. A knowledge chunk kc provides a synthetic representation of an element $e \in \mathcal{O}$ in terms of its constituent axioms/assertions, both explicitly and implicitly defined in $\mathcal{O}$. To this end, kc is defined as a set of axioms $kc = \{a_1(kc), a_2(kc), \ldots, a_n(kc)\}$ constituting the specification of the corresponding element e. An axiom $a_i(kc)$, with $i \in$ [1,n] has the form $a_i(kc) = \langle n(kc), r(a_i), v(a_i), id \rangle$ where:

- $n(kc) \in \mathcal{N}$ is the name of the knowledge chunk kc, which coincides with the name of e.
- $r(a_i) \in \mathcal{P} \cup \{\equiv, \sqsubseteq\}$ is a semantic relation contained in the definition of e.
- $v(a_i) \in \mathcal{N} \cup \mathcal{L}$ is the value of the corresponding relation $r(a_1)$.
- id is the provenance of kc, namely the identifier of the resource from which kc is generated.

Given $\mathcal{O}$, a set of knowledge chunks is derived to provide a synthetic representation of concepts and instances contained in $\mathcal{O}$. In particular, a knowledge chunk kc_u is created for each URI $u \in \mathcal{O}$. An axiom $a_i(kc_u)$ is defined for each path $u \rightarrow v$ between u and a terminal node v in the RDF graph of the resource. The semantic relation $r(a_i)$ is defined as the concatenation of the labels of the properties p_1 and p_n in $u \rightarrow v$, while the value $v(a_i)$ is set to v. An example of knowledge chunk is given in Section 3.2, while a detailed description of the construction of knowledge chunks from RDF(S) and OWL resources is provided in [17].

Mapping. A mapping m denotes a semantic correspondence between two knowledge chunks kc_i and kc_j and it is a tuple of the form $m = \langle n(kc_i), n(kc_j), SA, \mathcal{U} \rangle$, where:

- $n(kc_i)$ and $n(kc_j)$ are the names of the matching knowledge chunks kc_i and kc_j, respectively.
- $SA \in [0, 1]$ is the semantic affinity value denoting the level of matching between kc_i and kc_j.
- $\mathcal{U}$ is a (possibly empty) set of mapping rules, each one denoting the correspondence between pairs of matching axioms of kc_i and kc_j.

Context. A context defines a variability level in terms of operations on knowledge chunks and on mappings between them. The idea of using contexts to model matching variability derives from the field of conceptual modeling where

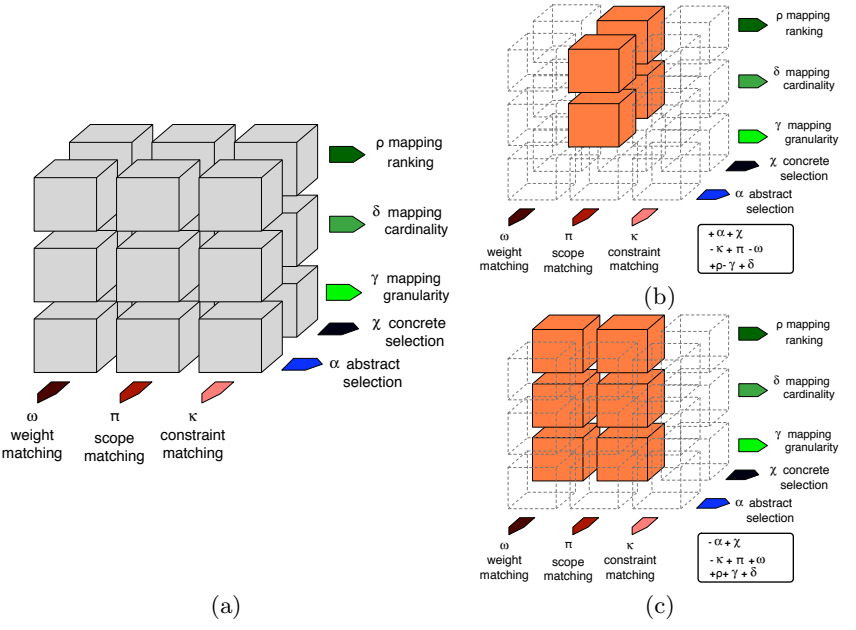


Fig. 1. Graphical representation of variability dimensions of matching (a), and examples of matching configurations (b) (c)

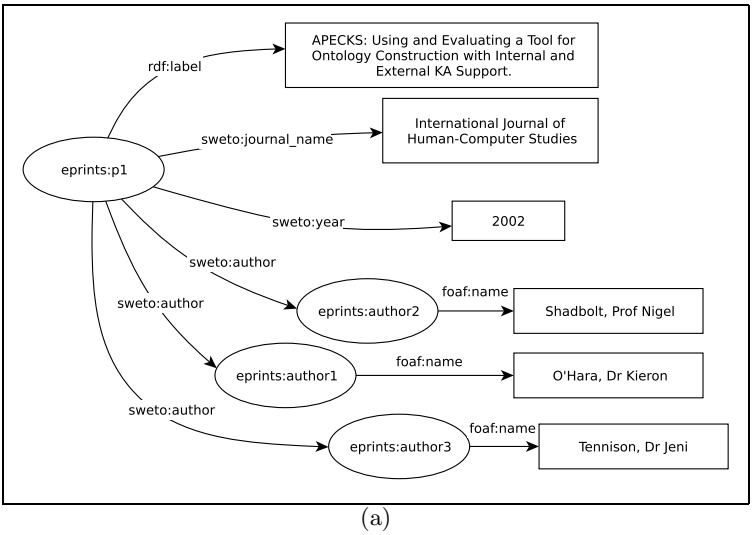
AKT EPrints

eprints:p1	**sweto:author**: O'Hara, Dr Kieron, Shadbolt, Prof Nigel, Tennison, Dr Jeni **rdf:label**: APECKS: Using and Evaluating a Tool for Ontology Construction with Internal and External KA Support. **sweto:journal_name**: International Journal of Human-Computer Studies **sweto:year**: 2002
eprints:p2	**sweto:author**: Alani, Dr Harith, Dasmahapatra, Dr Srinandan, Gibbins, Dr Nicholas, Glaser, Hugh, Harris, Steve, Kalfoglou, Dr Yannis, O'Hara, Dr Kieron, Shadbolt, Prof Nigel **rdf:label**: Managing Reference: Ensuring Referential Integrity of Ontologies for the Semantic Web. **sweto:book_title**: 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW'02) **sweto:year**: 2002
eprints:p3	**sweto:author**: Carr, Dr Leslie, Kampa, Dr Simon, Miles-Board, Mr Timothy **rdf:label**: Hypertext in the Semantic Web. **sweto:book_title**: ACM Conference on Hypertext and Hypermedia 2001 **sweto:year**: 2001 **sweto:pages**: pp. 237-238

Rexa

rexa:p1	**sweto:author**: Kieron O'Hara, Nigel R. Shadbolt **sweto:journal_name**: International Journal of Human-Computer Studies **sweto:year**: 2002
rexa:p2	**sweto:author**: Harith Alani, Srinandan Dasmahapatra, Nicholas Gibbins, Hugh Glaser **rdf:label**: Ensuring referential integrity of ontologies for the semantic web. **sweto:book_title**: Managing reference **sweto:year**: 2002 **sweto:pages**: 317-334

Fig. 2. A portion of the AKT EPrints and of the Rexa datasets



(a)

kc_i	$r(a_i)$	$v(a_i)$	id
eprints:p1	author_name	Shadbolt, Prof Nigel	eprints
	author_name	Tennison, Dr Jeni	eprints
	author_name	O'Hara, Dr Kieron	eprints
	label	APECKS: Using and Evaluating a Tool for Ontology Construction with Internal and External KA Support.	eprints
	journal_name	International Journal of Human-Computer Studies	eprints
	year	2002	eprints

(b)

Fig. 3. Example of RDF description of a publication (a) and the corresponding knowledge chunk (b)

it has been introduced for the purpose of dealing with domain variability and requirements engineering [18,19].

In the Matching Variability Framework, we abstract a matching execution $Match$ as $Match(\mathcal{D}) \rightarrow \mathcal{M}$, where $\mathcal{D}$ is a *source-dataset*, namely the set of knowledge chunks considered for the matching execution, and $\mathcal{M}$ is a *mapping-set*, namely the set of mappings produced as a result of the matching execution.

3.2 Matching Example

To classify matching variability and to illustrate our proposed framework, we consider bibliographic data based on the ARS benchmark datasets provided by the instance matching track of OAEI 2009[1].

In particular, we focus on a test case that includes two datasets in the domain of scientific publications, namely *AKT EPrints* and *Rexa*[2]. A portion of the

[1] Ontology Alignment Evaluation Initiative: http://oaei.ontologymatching.org/2009

[2] Both the datasets are available at http://www.intancematching.org/

test case is shown in Figure 2 where we report data extracted from the RDF description of three publications from AKT EPrints and two publications from Rexa, respectively. An example of the RDF representation for the publication eprints:p1 (see Figure 2) is shown in Figure 3(a) according to the SWETO-DBLP ontology[3], an extension of FOAF[4] for the domain of scientific publications and related authors. In particular, the properties *rdf:label* and *sweto:author* are used to represent the title and the author of publications, respectively. An example of the knowledge chunk corresponding to the bibliographic record eprints:p1 is shown in Figure 3(b).

In the following sections, we introduce the contexts that formalize the variability levels along each dimension of Figure 1.

4 The Source-Dataset Contexts

Variability along the source-dataset dimension is formalized through the following two contexts.

Definition 1. Abstract context. *The abstract context of a knowledge chunk kc is a unary operation $\alpha_c(kc) \rightarrow 2^{kc}$ that returns the set of axioms $\overline{kc} \subseteq kc$ that satisfy the abstract context condition c associated with $\alpha_c(kc)$. The condition c is an arbitrary combination of conjunctions and/or disjunctions of property names $p \in \mathcal{P}$. Each component of the condition c is satisfied if $\exists r(a_i) \in kc \mid r(a_i) = p$.*

Abstract contexts filter data with respect to the data structure, that is the set of properties and relations associated with a given knowledge chunk. For example, in order to focus only on journal papers and their titles for matching publications, we apply the abstract context $\alpha_{journal\_name \wedge label}$ to the knowledge chunks of eprints in Figure 2. As result, we have that only eprints:p1 satisfies the abstract condition and will be included in the matching operation (see Figure 4).

kc_i	$r(a_i)$	$v(a_i)$	id
eprints:p1	label	APECKS: Using and Evaluating a Tool for Ontology Construc- tion with Internal and External KA Support.	eprints
	journal_name	International Journal of Human-Computer Studies	eprints

Fig. 4. Example of abstract context result

Definition 2. Concrete context. *The concrete context of a knowledge chunk kc is a unary operation $\chi_c(kc) \rightarrow \{kc, \emptyset\}$ that returns the knowledge chunk kc itself if the axioms of kc satisfy the concrete context condition c, and returns the empty set otherwise. The concrete context condition c is an arbitrary combination of conjunctions and/or disjunctions of boolean predicates of the form $\langle v(a_i)\ \theta\ k \rangle$, where k is a constant value, and θ is a comparison operator in $\{>, <, =, \geq, \leq, \neq, LIKE\}$, where $LIKE$ denotes a pattern matching operator for strings (such as in SQL).*

[3] http://lsdis.cs.uga.edu/projects/semdis/swetodblp/august2007/opus_august2007.rdf
[4] http://xmlns.com/foaf/spec/

Concrete contexts filter source data with respect to their contents, that is the property values associated with a given knowledge chunk. Still referring to publications, we would focus the matching execution only on the publications produced after 2001. We apply the concrete context $\chi_{year > 2001}$ to eprints publications of Figure 2, by selecting only eprints:p1 and eprints:p2.

5 The Matching-Dataset Contexts

Variability along the matching-dataset dimension is formalized through three specific contexts as follows.

Definition 3. Constraint context. *The constraint context is a binary operation* $\kappa_c(kc, kc') \rightarrow \{(kc, kc'), \emptyset\}$ *that, given two knowledge chunks* kc *and* kc' *submitted to matching, returns the pair* (kc, kc') *if* kc *and* kc' *satisfy the constraints* c*, and returns the empty set otherwise. The constraint* c *is an arbitrary conjunction/disjunction of boolean predicates of the form* $\langle p\ \theta\ p' \rangle$*, where* θ *is a comparison operator in* $\{>, <, =, \geq, \leq, \neq\}$ *and* p *and* p' *denote two property names in* $\mathcal{P}$*, respectively. The predicate* $p\ \theta\ p'$ *is satisfied if* $\exists a_i(kc), a_j(kc') \mid r(a_i) = p, r(a_j) = p', (v(a_i)\ \theta\ v(a_j)) = \texttt{true}$.

A constraint context defines pre-condition(s) that must be satisfied by two knowledge chunks submitted to matching in order to be further considered for the purpose of matching. Conditions under which two knowledge chunks are considered as comparable (and thus can be further matched according to the scope and weight contexts) are expressed by the constraints in c. A very common constraint that could be required is the equality constraint $p = p'$, stating that two knowledge chunks kc_i and kc_j are equality-comparable only if their properties p and p' have the same value. For example, with respect to publications of Figure 2, if we want to match only records of publications appeared in the same year, we apply the constraint context $\kappa_{kc_i.year=kc_j.year}(kc_i, kc_j)$ to pairs of knowledge chunks in (AKT EPrints × Rexa), resulting in the following set $\overline{KC}$ of comparable knowledge chunk pairs.

$$\overline{KC} = \{(eprints : p1, rexa : p1), (eprints : p1, rexa : p2),$$
$$(eprints : p2, rexa : p1), (eprints : p2, rexa : p2)\}$$

Definition 4. Scope context. *The scope context is defined as a binary operation* $\pi_c(kc, kc')$ *that, given a pair of knowledge chunks* kc *and* kc' *returns the scope-projection* $(\overline{kc}, \overline{kc'})$ *of* kc *and* kc' *under the scope* c*, with* $c \in \{\texttt{terminological}, \texttt{structural}, \texttt{full}\}$. *The scope-projection* $(\overline{kc}, \overline{kc'})$ *is defined according to the following rules:*

- *If* $c = \texttt{terminological}$ *then*
  - $\overline{kc} = \{n_i \mid \exists a_i(kc), r(a_i) = n_i \vee \exists a_j(kc), v(a_j) = n_i\}$
  - $\overline{kc'} = \{n'_i \mid \exists a_i(kc'), r(a_i) = n'_i \vee \exists a_j(kc'), v(a_j) = n'_i\}$

- *If* $c =$ `structural` *then*
  - $\overline{kc} = \{\langle n_j, r(a_j)\rangle \mid \exists a_i(kc), r(a_i) = r(a_j), n(kc) = n_j\}$
  - $\overline{kc'} = \{\langle n'_j, r(a_j)'\rangle \mid \exists a_i(kc'), r(a_i) = r(a_j)', n(kc') = n'_j\}$
- *If* $c =$ `full` *then*
  - $\overline{kc} \equiv kc$
  - $\overline{kc'} \equiv kc'$

The scope context has the purpose of keeping only the features of the knowledge chunk representation that are of interest for comparison and matching evaluation. In particular, the `terminological` scope limits to the terms appearing in the knowledge chunk. In the Matching Variability Framework, we call this set of terms *terminological equipment* of a knowledge chunk, which represents the (unstructured) terminological information available in a knowledge chunk. The `structural` scope produces the set of properties of kc and kc' by cutting off their values. In this case, only the knowledge chunk structure is considered during matching. Finally, the `full` scope considers all the information available in a knowledge chunk. As an example, if we are interested in matching the publications of Figure 2 eprints:p1 and rexa:p1 on the basis of their structure only, we apply the scope context $\sigma_{structural}(eprints : p1, rexa : p1)$ that returns the scope-projection $\overline{eprints : p1} = \{\langle eprints : p1, author\_name\rangle, \langle eprints : p1, label\rangle, \langle eprints : p1, journal\_name\rangle, \langle eprints : p1, year\rangle\}$ and $\overline{rexa : p1} = \{\langle rexa : p1, author\_name\rangle, \langle rexa : p1, journal\_name\rangle, \langle rexa : p1, year\rangle\}$.

Definition 5. Weight context. *The weight context* $\omega_c(kc) \rightarrow \overline{KC}$ *is defined as a unary operation that, given a knowledge chunk kc returns a weighted knowledge chunk* $\overline{kc}$ *defined according to the weighting set c. The weighting set c is composed by pairs of the form* (p_i, w_i)*, where* p_i *is a property name in* $\mathcal{P}$ *and* w_i *is a weight in the range [0,1]. For each weighting pair* (p_i, w_i)*, the resulting weighted knowledge chunk* $\overline{kc}$ *is defined as* $\overline{kc} = \{\langle a_i(\overline{kc}), w_i\rangle \mid \exists a_i(kc) = a_i(\overline{kc}) \wedge r(a_i) = p_i\}$.

The weight context allows to discriminate the relevance to be assigned to axioms for knowledge chunk comparison. The higher the weight is, the highest the relevance of the axiom is. An important usage of the weight context is to assign more relevance to axioms having capability of identifying objects. Axioms having strong identification power can be set to have higher relevance with respect to the others in determining the final level of matching. For example, considering publications of Figure 2, we want to set titles and authors as more relevant properties for identification than book titles and years, while pages should not be taken into account at all. To this end, we define the weight context condition (author_name, 1.0), (label, 1.0), (book_title, 0.5), (year, 0.5), (pages, 0.0).

6 The Mapping-Set Contexts

Variability along the mapping-set dimension is addressed by the following contexts.

Definition 6. Ranking context. *The ranking context* $\rho_c(\mathcal{M}) \rightarrow 2^{\mathcal{M}}$ *is defined as a unary operation that, given a mapping-set* $\mathcal{M}$ *returns a ordered list* $\overline{\mathcal{L}}$ *of mappings of* $\mathcal{M}$ *filtered according to the threshold* $c \in$ *[0,1]. In particular,* $\overline{\mathcal{L}}$ *is defined as follows:*

$$\overline{\mathcal{L}} = (m_0, m_1, \ldots, m_n) \mid \forall m_i, m_j, j > i \Rightarrow SA_j \geq SA_i \geq c$$

The ranking context is used to cut off mappings whose level of semantic affinity is lower than a given threshold. For example, referring to publications of Figure 2, if we execute matching of eprints against rexa by applying the abstract context α_{author_name}, we obtain the following mappings:

$$m_1 = \langle eprints : p1, rexa : p1, 0.8, \emptyset \rangle$$
$$m_2 = \langle eprints : p2, rexa : p1, 0.4, \emptyset \rangle$$
$$m_3 = \langle eprints : p2, rexa : p2, 0.67, \emptyset \rangle$$

This result can be refined by cutting off mappings with a low semantic affinity value, by applying the ranking context $\rho_{0.5}$ that returns only the mappings m_1 and m_3.

Definition 7. Cardinality context. *The cardinality context* $\delta_c(\mathcal{M}) \rightarrow 2^{\mathcal{M}}$ *is defined as a unary operation that, given a mapping-set* $\mathcal{M}$ *returns a mapping-set* $\overline{\mathcal{M}} \subseteq \mathcal{M}$ *that contains only mappings compatible with the cardinality constraint* $c \in \{\texttt{one}, \texttt{many}\}$*. The resulting mapping-set* $\overline{\mathcal{M}}$ *is defined according to the following rules:*

- $c =$ `one`*:* $\overline{\mathcal{M}} = \{m_i = \langle n(kc_i), n(kc'), SA_i, \mathcal{U}_i \rangle \mid \exists! kc', \langle n(kc_i), n(kc'), SA_i, \mathcal{U}_i \rangle$
- $c =$ `many`*:* $\overline{\mathcal{M}} \equiv \mathcal{M}$

The cardinality context regulates the maximum number of target knowledge chunks that can match a single source knowledge chunk in a mapping-set. This number (called cardinality) ranges from the value 'unbounded' allowing to keep all the mappings discovered during the matching execution to 'exactly one', which takes only the "best matching" mapping, where the notion of best matching is defined according to the requirements of the matching task at hand. With respect to the previous example, if we do not apply any ranking context, the knowledge chunk eprints:p2 matches with both rexa:p1 and rexa:p2. This situation can be handled by applying the cardinality context $\delta_{\texttt{one}}$ and by selecting as best matching element the one with the highest semantic affinity. This results in keeping only the mapping $\langle eprints : p2, rexa : p2, 0.67, \emptyset \rangle$.

Definition 8. Granularity context. *The granularity context* $\gamma_c(\mathcal{M}) \rightarrow 2^{\mathcal{M}}$ *is a unary operation that, given a mapping-set* $\mathcal{M}$ *returns a mapping-set* $\overline{\mathcal{M}} \subseteq \mathcal{M}$ *defined according to the granularity condition* $c \in \{\texttt{simple}, \texttt{complex}\}$*. The resulting mapping-set* $\overline{\mathcal{M}}$ *is defined according to the following rules:*

- $c =$ `simple`: $\overline{\mathcal{M}} = \{m_i = \langle n(kc), n(kc'), SA_i, \mathcal{U}_i \rangle \mid m_i \in \mathcal{M}, \mathcal{U}_i = \emptyset$
- $c =$ `complex`: $\overline{\mathcal{M}} = \{m_i = \langle n(kc), n(kc'), SA_i, \mathcal{U}_i \rangle \mid m_i \in \mathcal{M}, \mathcal{U}_i \neq \emptyset$

The granularity context determines the kind of mapping that holds between two knowledge chunks in a mapping-set, ranging from a simple correspondence at the whole knowledge chunk level to more complex mappings specifying also mapping rules, which state how to transform the axioms of one knowledge chunk into the matching ones of the other knowledge chunk. As an example we consider a mapping m_i between eprints:p2 and rexa:p2. A complex granularity mapping specifies the following mapping rules: $(eprints : p2).author\_name \Leftrightarrow (rexa : p2).author\_name$, $(eprints : p2).label \Leftrightarrow (rexa : p2).book\_title + (rexa : p2).label$, $(eprints : p2).book\_title \Leftrightarrow NULL$, $(eprints : p2).year \Leftrightarrow (rexa : p2).year$, $NULL \Leftrightarrow (rexa : p2).pages$. The mapping rules state that the label of eprints:p2 corresponds to the concatenation of book_title and label of rexa:p2; the book_title in eprints:p2 does not have a corresponding value in rexa:p2; pages in rexa:p2 does not have a corresponding value in eprints:p2.

7 Matching Semantic Web Data with Contexts

In this section, we show an instantiation of the Matching Variability Framework in HMatch 2.0 [20]. HMatch 2.0 is developed as a flexible matching suite where a number of matching techniques are implemented and organized in different modules providing linguistic (HMatch(L)), concept (HMatch(C)), and instance (HMatch(I)) matching techniques. These HMatch 2.0 modules can be differently combined to provide four matching models, namely *surface*, *shallow*, *deep*, and *intensive*, which allows the implementation of the contexts along the matching-dataset variability dimension. Finally, the mapping-set contexts can be realized by proper configuration of HMatch(M), the mapping-manager module of HMatch 2.0. As an example of matching semantic web data with contexts, we report the HMatch 2.0 performance in the OAEI 2009 instance matching contest, where we exploited HMatch 2.0 for matching the whole AKT Eprints and Rexa sources. Goal of this matching execution was to find the bibliographic records referred to the same real publications between 2000 and 2003. This conceptual target corresponds to the contexts $\langle -\alpha, +\chi, -\kappa, +\pi, +\omega, +\rho, +\delta, +\gamma \rangle$, which has been obtained with the combination of the properly configured HMatch 2.0 techniques/modules shown in Figure 5.

As a first step, we translated the original RDF datasets into a collection of knowledge chunks $\mathcal{D}$. The KC-Wrap tool embeds functionalities for the derivation of knowledge chunks from OWL ontologies and RDF repositories. Moreover, KC-Wrap also implements the abstract and concrete contexts defined in our Matching Variability Framework. We exploited a concrete context in order to limit the matching task to data referred to the years between 2000 and 2003. As a result of this step, the source-dataset $\mathcal{D}$ containing all the records in AKT Eprints and Rexa has been transformed into a smaller dataset $\overline{\mathcal{D}}$ (i.e., matching-dataset) containing only the bibliographic records of interest for the considered matching

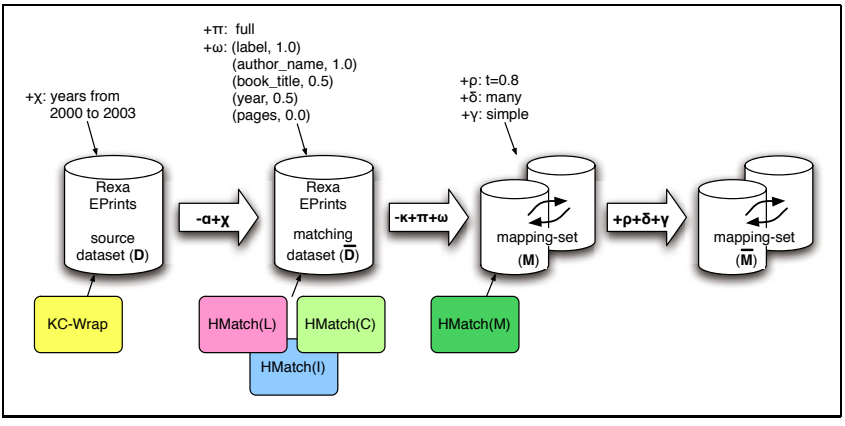


Fig. 5. Instantiation of the Matching Variability Framework in HMatch 2.0

target. Then, we exploited a `full` scope context, corresponding to the intensive matching model of HMatch 2.0, and a weight context in order to configure the matching process. In particular, the full scope-projection has been chosen in order to take into account all the information available about bibliographic records, while a weight context has been enforced to set properties `label` and `author_name` as the most relevant for object identification, followed by `year` and `book_title` (see Figure 5). The value of the property `pages` has not been taken into account, by setting its weight to 0.0. According to this configuration, the matching process is executed, leading to a resulting mapping-set $\mathcal{M}$. Then, HMatch(M) is exploited to select the mappings with a semantic affinity value greater then or equal to 0.8 (i.e., mapping ranking context). Moreover, we adopted a `many` cardinality context and a `simple` granularity context. The cardinality represents the fact that we can have more than one record representing the same publication in the original source-dataset, while the choice of the granularity context was indicated by OAEI 2009 regulations. The resulting mapping-set $\overline{\mathcal{M}}$ contains the bibliographic records referred to the same real publications and it has been validated against the set of expected mapping provided by OAEI 2009, obtaining a precision of 0.95 and a recall of 0.46, that is the second best performance of OAEI 2009[5].

8 Concluding Remarks

In this paper, we discussed the notion of matching variability and we presented the Matching Variability Framework for its formal representation and classification. An example of instantiation of this framework in our matching system HMatch 2.0 has been described by considering a test case of bibliographic records of OAEI 2009. Ongoing work is mainly devoted to the integration in the HMatch

[5] http://islab.dico.unimi.it/content/oaei2009

2.0 system of the wrapping tools and related contexts at the source-dataset level. Moreover, we plan to use the Matching Variability Framework for a comparative analysis of recently developed instance matching tools.

References

1. Nikolov, A., Uren, V., Motta, E., Roeck, A.D.: Handling Instance Coreferencing in the KnoFuss Architecture. In: Proc. of the 1st ESWC Int. Workshop on Identity and Reference on the Semantic Web (IRSW 2008), Tenerife, Spain (2008)
2. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. The VLDB Journal 10(4), 334–350 (2001)
3. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer, Heidelberg (2007)
4. Isaac, A., van der Meij, L., Schlobach, S., Wang, S.: An Empirical Study of Instance-Based Ontology Matching. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 253–266. Springer, Heidelberg (2007)
5. Bleiholder, J., Naumann, F.: Data Fusion. ACM Computing Surveys 41(1) (2008)
6. Shvaiko, P., Euzenat, J.: Ten Challenges for Ontology Matching. In: Meersman, R., Tari, Z. (eds.) OTM 2008, Part II. LNCS, vol. 5332, pp. 1164–1182. Springer, Heidelberg (2008)
7. Jean-Mary, Y.R., Shironoshita, E.P., Kabuka, M.R.: Ontology Matching with Semantic Verification. Web Semantics: Science, Services and Agents on the World Wide Web 7(3), 235–251 (2009)
8. Li, J., Tang, J., Li, Y., Luo, Q.: RiMOM: A Dynamic Multistrategy Ontology Alignment Framework. IEEE Transactions on Knowledge and Data Engineering 21(8), 1218–1232 (2009)
9. Nagy, M., Vargas-Vera, M., Motta, E.: Managing Conflicting Beliefs with Fuzzy Trust on the Semantic Web. In: Gelbukh, A., Morales, E.F. (eds.) MICAI 2008. LNCS (LNAI), vol. 5317, pp. 827–837. Springer, Heidelberg (2008)
10. Doshi, P., Thomas, C.: Inexact Matching of Ontology Graphs Using Expectation-Maximization. In: Proc. of the 21st National Conference on Artificial Intelligence (AAAI 2006), Boston, Massachusetts, pp. 1277–1282 (2006)
11. Cruz, I.F., Antonelli, F.P., Stroe, C.: AgreementMaker: Efficient Matching for Large Real-World Schemas and Ontologies. In: Proc. of the 35th Int. Conference on Very Large Data Bases (VLDB 2009), Lyon, France, pp. 1586–1589 (2009)
12. Zimmermann, A., Krötzschand, M., Euzenat, J., Hitzler, P.: Formalizing Ontology Alignment and its Operations with Category Theory. In: Proc. of the 2006 Conference on Formal Ontology in Information Systems, Amsterdam, The Netherlands, pp. 277–288 (2006)
13. Euzenat, J.: Algebras of Ontology Alignment Relations. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 387–402. Springer, Heidelberg (2008)
14. Gal, A., Martinez, M.V., Simari, G.I., Subrahmanian, V.S.: Aggregate Query Answering under Uncertain Schema Mappings. In: Proc. of the IEEE Int. Conference on Data Engineering (ICDE 2009), Washington, DC, USA, pp. 940–951 (2009)
15. Meilicke, C., Völker, J., Stuckenschmidt, H.: Learning Disjointness for Debugging Mappings between Lightweight Ontologies. In: Gangemi, A., Euzenat, J. (eds.) EKAW 2008. LNCS (LNAI), vol. 5268, pp. 93–108. Springer, Heidelberg (2008)

16. Castano, S., Ferrara, A., Lorusso, D., Näth, T.H., Möller, R.: Mapping Validation by Probabilistic Reasoning. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 170–184. Springer, Heidelberg (2008)
17. Castano, S., Ferrara, A., Montanelli, S.: The iCoord Knowledge Model for P2P Semantic Coordination. In: Proc. of the 6th Conference of the Italian Chapter of AIS, Costa Smeralda (Nu), Italy (2009)
18. Lapouchnian, A., Mylopoulos, J.: Modeling Domain Variability in Requirements Engineering with Contexts. In: Proc. of the 28th Int. Conference on Conceptual Modeling (ER 2009), Gramado, Brazil. Springer, Heidelberg (2009)
19. Van Lamsweerde, A.: Goal-Oriented Requirements Engineering: A Guided Tour. In: Proc. of the 5th IEEE Int. Symposium on Requirements Engineering (RE 2001), Washington, DC, USA (2001)
20. Castano, S., Ferrara, A., Montanelli, S.: Matching Ontologies in Open Networked Systems: Techniques and Applications. Journal on Data Semantics, JoDS V (2006)