Towards Data Fusion in a Multi-ontology Environment

Andriy Nikolov
a.nikolov@open.ac.uk

Victoria Uren
v.s.uren@open.ac.uk

Enrico Motta
e.motta@open.ac.uk

Knowledge Media Institute
Open University
Milton Keynes, UK

ABSTRACT

With the growing amount of semantic data being published on the Web the problem of finding individuals in different datasets which correspond to the same entity is gaining importance. Given that datasets are often structured using different ontologies, automatic schema-matching techniques have to be utilized before proceeding with data-level alignment. In this paper we discuss how ontology schema mismatches influence data-level alignment based on our first experience with implementing a data fusion tool for a multi-ontology environment.

Categories and Subject Descriptors

H.4.m [**Information Systems**]: Miscellaneous;
D.2 [**Software**]: Software Engineering

Keywords

Data fusion, coreference resolution, linked data

1. INTRODUCTION

The data integration process has to deal with two top-level problems: resolving schema-level and data-level issues. On the Web scale, semantic heterogeneity of data is inevitable, which makes it necessary for a data coreference resolution system to use results of automatic ontology matching techniques. These techniques do not guarantee 100% accuracy and errors produced by them may influence the quality of the data fusion stage. In our previous work we developed an architecture for semantic data fusion called KnoFuss [14]. The initial version of the system was designed for the enterprise knowledge management scenario, in which it was assumed that schema-level issues were resolved and datasets being integrated were already structured according to the same ontology. We implemented an extension of the system, which utilizes schema-level mappings, produced automatically, to resolve coreferences between datasets using different ontologies. In this paper we discuss the impact of the ontology heterogeneity on the quality of instance coreferencing.

2. ONTOLOGICAL MISMATCHES AND DATA INTEGRATION ISSUES

The situation when datasets to be integrated use different ontologies makes it hard for data integration methods to use the semantic data structure. Mappings between ontology terms are needed to provide a uniform view over individuals in two datasets and make the individuals comparable.

2.1 Ontological mismatches and correspondence patterns

Obtaining an adequate representation of mappings which allows correct data transformation is a non-trivial problem due to ontology mismatches. A classification framework of different types of mismatches between overlapping ontologies was given in [11]. Assuming that ontologies are represented in the same language, the framework distinguishes:

- *Conceptualisation mismatches* caused by different ways of domain interpretation. These different ways in turn may concern:
  - Scope, when two classes seemingly representing the same concept do not contain the same instances (e.g., the class *PoliticalOrganization* in TAP ontology includes terrorist groups, while in SWETO it is meant to represent only legal organisations).
  - Model coverage and granularity, when parts of the domain in one ontology are not covered in another or covered with a different level of detail (e.g., in SWETO the class *Company* does not have subclasses while TAP and DBPedia 3.2 distinguish between different types of companies).
- *Explication mismatches* caused by different ways the conceptualisation is specified. These are further divided into:
  - Modelling style mismatches, when the same domain is modeled using different paradigms (e.g., point vs interval logic for time representation) or concept specification (e.g., splitting the subclasses of the same class in a hierarchy according to different criteria).
  - Terminological mismatches, when different terms are used to represent the same entity (synonymy) or the same term represents different entities (homonymy).
  - Encoding mismatches, when the values at the data level have different formats. This one has to be dealt at the data-level stage, so we do not consider it in this paper.

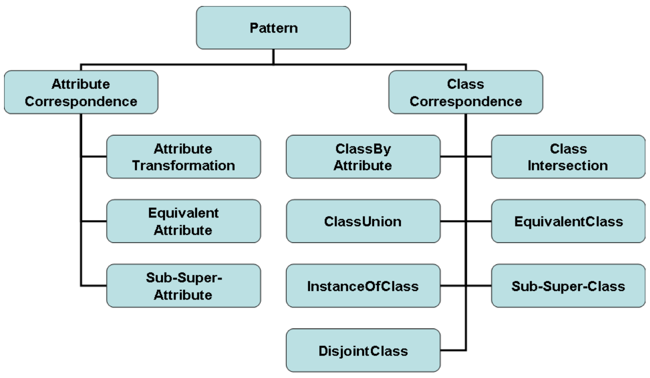

Figure 1: Correspondence patterns of ontology matching according to [16] (fragment). A commonly used *DisjointClass* pattern is included.

To represent correctly the correspondences between ontologies and overcome these mismatches mappings of varying degrees of complexity are required. In [16] common correspondence patterns are introduced to represent such mappings (see Fig. 1). For the most part mapping patterns represent description logic relations. Available automatic ontology matching algorithms can only produce a subset of possible mappings. Given the limited capabilities of ontology matching tools we can expect that some of the ontology mismatches will remain unresolved or partially unresolved at the data integration stage. Below we try to consider the impact of such mismatches during the data integration process.

2.2 Data-level impact of ontology mismatches

The first type of mismatches in the classification presented in [11] concerns conceptualisation. For the coreference resolution stage shared conceptualisation allows the system to:

- consider individuals belonging to the same class as candidates for matching;
- estimate the likelihood of individuals being equivalent given available evidence (e.g., having two people with the same name belonging to a specific class *SemanticWebResearcher* is a much stronger evidence of equivalence than if they only had a generic class *Person* in common).

Conceptualisation mismatches between two ontologies (in particular, scope mismatches) may reduce both recall and precision of coreference resolution algorithms. For example, the class *Company* in SWETO does not include financial organisations, while its counterpart in TAP includes them. Thus, when the system tries to find for each company in TAP coreferent individuals in SWETO only having the equivalence relation between these classes, it will not find matching pairs for financial organisations, because they belong to a different class in SWETO. This will make the recall decrease. On the other hand, the class *ComputerScientist* in TAP contains only world-famous computer scientists while most researchers are classified according to their place of work (e.g., *CMUPerson*, *W3CPerson*). *ComputerScienceResearcher* in SWETO, which automatic tools often consider equivalent, has much wider coverage and includes

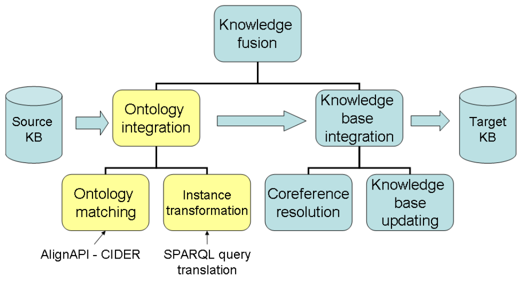

Figure 2: Fusion task decomposition incorporating schema matching.

everybody who contributed to a CS paper mentioned in the knowledge base. Thus, labels in SWETO are much more ambiguous and the danger of matching two unrelated individuals increases, which may affect precision. The same happens when there is no equivalence between classes but a *Sub-Super-Class* relation: the same degree of similarity between individuals may provide much weaker evidence, which makes it hard to adequately estimate the reliability of methods' output. Another area of impact involves disjointness relations. Disjointness between classes can be used as evidence to consider some coreference mappings incorrect and delete them. Scope mismatches can lead to errors when classes considered disjoint in one ontology are overlapping in another one (like in the case with *PoliticalOrganization* and *TerroristOrganization* above): correct mappings can be deleted if they are perceived as causing inconsistency. Granularity mismatches do not allow using ontological constraints defined for classes at the lower levels of the hierarchy if the other ontology does not distinguish between these classes.

Among the explication mismatches modelling style differences are the hardest to solve automatically. Translation between paradigms is a very domain-specific problem and common correspondence patterns are often not sufficient to align two ontologies. In a simple example case, if one ontology represents colours using a set of pre-defined labels (red, yellow, black) and another one uses RGB encoding, it is very hard to find similar values automatically: a hand-tailored matching procedure is necessary. To our knowledge, no existing automatic ontology matching tool is capable of dealing with different paradigms. For the case when subclasses of the same class in two ontologies are split according to different criteria, no useful DL relations can be established between them (apart from the fact that there may be some overlap). Such differences can make any automatic data integration procedures intractable. If these mismatches occur at lower levels of the hierarchy, methods can operate only with information defined at a higher level.

Finally, terminological mismatches are the primary focus of most existing ontology matching tools [5], which makes them the simplest to handle. They can be solved by creating *EquivalentClass* and *EquivalentAttribute* correspondences.

3. KNOFUSS ARCHITECTURE

The KnoFuss architecture [14] implements a modular framework for semantic data fusion. The fusion process is divided into subtasks as shown in the Fig. 2 and the architecture focuses on its second stage: knowledge base integration. The first subtask is coreference resolution: finding potentially coreferent instances based on their attributes. The next stage, knowledge base updating, refines coreferencing results taking into account ontological constraints, data conflicts and links between individuals. Algorithms performing fusion subtasks (e.g., string-based similarity matchers) are represented as problem-solving methods. All methods for the same task have a common interface and their capabilities (range of applicability and reliability of output) are formally defined using the fusion ontology. Because each algorithm behaves differently depending on the data to which it is applied, optimal parameters can be defined depending on the *application context* (type of data): e.g., Jaro-Winkler string similarity is appropriate for comparing person names but not suitable for publication titles, etc.

To deal with the multi-ontology scenario the architecture has to cover the ontology integration stage, which includes two subtasks: ontology matching and instance transformation.

3.1 Ontology matching

The *Ontology matching* task involves creation of mapping rules or alignments: sets of correspondences between two ontologies [5].

Considering correspondence patterns, data fusion needs both correspondences between concepts (*ClassCorrespondence*) and correspondences between properties (*AttributeCorrespondence*). Class mappings allow relevant method application contexts to be translated into the terms of the source ontology, if they were initially defined in terms of the target ontology. Attribute correspondences are needed in order to retrieve properties relevant for coreference resolution in both knowledge bases. Equivalence and subsumption relations allow relevant data structures in the source ontology to be found. Disjointness relations between concepts are usable for the *Knowledge base updating* stage, providing evidence for inconsistency resolution. The architecture assumes that ontology matching methods provide their output in the standard Alignment API format [4].

3.2 Instance transformation

The goal of the *Instance transformation* stage is to resolve structural differences between two knowledge bases so that the architecture itself and instance-level methods can process individuals in the source and target knowledge bases in the same way. Alignments produced by ontology matching methods are applied to provide a uniform view over data in two knowledge bases. In the KnoFuss architecture SPARQL queries are used as a primary means of retrieving data (method applicability ranges, application contexts, sets of relevant attributes). These queries are translated into the terms of the source ontology using available mappings. Sometimes a term in the target ontology potentially corresponds to several terms in the source ontology. This happens when there are several candidate *EquivalentClass* mappings provided by one or several ontology matching tools. In such situations we combine these mappings and consider them as a single *ClassUnion* mapping. For instance when we consider the query

SELECT *?uri* **WHERE** {
?uri rdf:type sweto:Computer_Science_Researcher }

the system tries to find all *ClassCorrespondence* mappings, which include the class *sweto:Computer_Science_Researcher.* In our example with the CIDER tool (see below) these included *EquivalentClass* mappings with classes *tap: CMUPerson*, *tap:ComputerScientist* and *tap:MedicalScientist.* Such a variety of potentially corresponding classes is caused by several existing mismatches between ontologies, in particular terminological mismatches (*Computer_Science_ Researcher* vs *ComputerScientist*), modelling style mismatches (*tap: CMUPerson* includes computer science researchers who worked in the CMU) and conceptualisation scope mismatches (*tap: ComputerScientist* represents only a subset of “world-famous” researchers and *tap:Medical-Scientist* includes authors of medical AI expert systems). From the strict logical point of view the only correct mapping would be a *Sub-Super-Class* mapping *tap:ComputerScientist* $\subseteq$ *sweto: Computer_Science_Researcher.* However, excluding other mappings would remove from consideration many TAP individuals, which have their equivalent SWETO counterparts. In reality, the data integration system needs information about partial alignments between concepts to select individuals which may potentially be coreferent rather than strict logical relations. We can call this the *OverlapClass* correspondence pattern. Thus, the query from our example is translated into:

SELECT *?uri* **WHERE**
{ {*?uri rdf:type tap:CMUPerson*}
UNION {*?uri rdf:type tap:Computer_Scientist*}
UNION {*?uri rdf:type tap:Medical_Scientist*}}

These pairs of queries assumed to be equivalent are then used at the later stages of the workflow, which allows the system to operate in the same way as in a single ontology case. At this stage the system utilizes the *DisjointClass* mappings. The system uses a simple algorithm to search for contradictory mappings: it finds situations when two classes in different ontologies are connected via a *Sub-Super-Class* mapping (created by ontology matching methods or inferred) and at the same time are disjoint (again, directly or via inference). Such mappings are considered conflicting. If the *DisjointClass* mapping has higher confidence then the contradictory *Sub-Super-Class* mapping (or the mapping it was inferred from) is removed from consideration.

4. EXPERIMENTS

To test the KnoFuss architecture in a multi-ontology scenario we used two artificially created knowledge bases intended to be used as benchmarks for Semantic Web applications: TAP [9] and SWETO testbed [1]. As primary methods for ontology matching we used two tools, which participated in the last OAEI contest: CIDER [8] and Lily [18]. Also we used the SCARLET service [15] as a method for generating *DisjointClass* mappings using existing ontologies defined elsewhere on the Web. Assuming that all sibling classes in the target ontology (SWETO) were mutually disjoint and using equivalence mappings produced by the CIDER tool we inferred additional disjointness mappings. Disjointness mappings were used to filter out conflicting equivalence relations with a low reliability. As coreference resolution methods for instances we used the same string similarity techniques as in our single-ontology scenario experiments [14]. While our experiments are still ongoing, from these tests we could make several observations.

First, as could be expected, errors during schema matching stage are propagated and can potentially lead to significant distortions during instance coreferencing. For instance, when matching instances of the class *sweto:Company* the

CIDER tool incorrectly aligned it with the class *tap:Country*. This led the coreference precision to drop to 41% while it reached 74% without this mistake (many companies have names derived from country names). We found ontological constraints to be extremely valuable as a means to repair such errors. Apart from the widely used *owl:FunctionalProperty* and *owl:InverseFunctionalProperty*, which allow non-ambiguous instance identification, ontological axioms, which may lead to inconsistency, allow filtering out incorrect mappings. These constraints include disjointness and datatype properties with cardinality constraints. E.g., knowing that *Company* is disjoint with *Country* (or inferring that) would repair the problem. However, most ontologies do not define these constraints explicitly because they are not needed in common ontology usage scenarios.

Second, although semantic heterogeneity (different meaning attached to similar resources) is seen primarily as a schema-level knowledge modelling issue, it can cause problems at the instance level as well. For instance, the TAP ontology contains a single individual describing the Coca-Cola Company while SWETO contains several individuals describing Coca-Cola branches in different countries. Whether such instances should be considered coreferent depends on the context of the task.

Then, as for the single-ontology scenario, it is hard to find a single instance matching algorithm to apply to all kinds of data: settings have to be optimized for a specific type of data rather than for a specific pair of ontologies as in schema matching. Ontology mismatches may lead not just to irrelevant instances being compared, but also to instances being compared using inappropriate similarity measures.

5. DISCUSSION

As we said in the beginning, our primary interest when implementing the version of the KnoFuss architecture to be used in a multi-ontology scenario was to observe the influence of schema-level mismatches on the data integration stage.

In comparison with the single-ontology data fusion scenario, adding the ontology heterogeneity challenge results both in decreased reliability of methods' output and difficulties in precise estimation of this decrease. For data-level coreference resolution methods we assume that the performance of the method depends on some common features of individuals belonging to a class: this assumption was the basis for the usage of application contexts in the KnoFuss architecture. For ontology matching methods even knowing the estimated quality of a method (e.g., precision/recall in some test scenarios) it is hard to estimate whether it will hold for a different pair of datasets. Second, it is hard to measure precisely the impact of a single ontology-level error at the data level. This possible negative impact can result in:

- Erroneous widening or narrowing of the applicability range of integration methods (misaligned concepts).

- Providing noisy evidence for data-level methods (misaligned properties and ontological restrictions).

Finally, some ontological mismatches, such as modelling style, cannot be resolved fully automatically by currently existing tools and can make data-level methods inapplicable. Based on our experience, we can outline several directions for assisting data fusion in the presence of schema heterogeneity.

First, label comparison is usually not considered sufficiently reliable evidence for coreference resolution (e.g., [7]). However, more complex algorithms utilizing context data (additional properties and links between individuals) can only be applied to datasets containing sufficiently overlapping data. It can be expected that many data integration tasks on the Web scale will only be able to rely on instance names and thus can only provide suggestions rather than generate *owl:sameAs* statements carrying strong implications. Given that the output is likely to be noisy it is necessary to keep track of data integration decisions (such as instance coreference mappings or statements considered incorrect) and their provenance. One possible way is to extend the coreference bundles approach [10] to include for each URI the confidence of its inclusion into the set.

Second, considering the limited capabilities of automatic ontology matching methods, availability of trusted reusable schema-level background knowledge is important. Such manually built reference knowledge is useful when it covers the gaps existing in common ontology matching scenarios. Among others, such reference knowledge may include:

- Specifying rich semantic restrictions existing in a certain domain, e.g., disjointness relations, property cardinality and domain/range constraints.

- Covering common ontological mismatches, which cannot be resolved automatically. For instance, these can include transformation rules between common time modelling approaches and overlaps between subclasses of the same concept divided according to different criteria (e.g., classifying historical artifacts from China by centuries or by dynastic periods). In this way a complex modelling style mismatch can be reduced to a terminological one, which can be treated automatically.

Third, sometimes existing automatic matching tools impose too rigid restrictions on their output aimed at improving the precision. For instance, some tools (like Lily) produce only one-to-one equivalence mappings assuming that two different classes in one ontology cannot be considered equivalent to the same class in another ontology. Thus, only the best candidate for equivalence is selected and all others are filtered out. While a useful assumption for terminological mismatches, it may miss important mappings in the presence of conceptualisation and modelling style mismatches. From the data fusion point of view it would be useful if ontology matching algorithms could produce weak mapping relations such as *ClassOverlap*.

6. RELATED WORK

Given the amount of data, which needs to be handled on the Web scale, the need to use automatic coreference resolution techniques is recognized in the Semantic Web community [2], [7], [6]. Among the existing systems Sindice [17] uses a straightforward method for coreference resolution by utilizing explicitly defined key properties (inverse functional properties). Individuals, which have equal values for such properties are considered equivalent. This is an approach which provides high precision but can only be applied to a limited subset of data, where such properties are defined explicitly and have values in a standard format. Other tools implement approximate matching techniques similar to those created in the database integration and ontology matching domains. The OKKAM server [3] used the Monge-Elkan string similarity metrics for selecting coreferent instances in the experiments. RDF-AI [12] concentrates on data-level issues when combining datasets using the same schema. The algorithm uses string (Monge-Elkan) and linguistic (WordNet) similarity to calculate distance between literal property values and then uses the iterative graph matching algorithm, similar to similarity flooding [13], to calculate distance between individuals.

7. SUMMARY AND FUTURE WORK

We implemented the first prototype of the KnoFuss data integration system for the multi-ontology environment and performed initial experiments with it. In our view, combining automatic schema-level and data-level alignment techniques in a single workflow still presents difficulties not only because schema-level matching tools occasionally produces errors, but also because some important types of ontology mismatches are not handled properly by them. In particular, this concerns conceptualisation and modelling style mismatches. While being very hard to solve automatically, there are several ways to assist the coreference resolution process when dealing with these mismatches, in particular:

- Extend the functionality of automatic schema-matching tools to discover different types of mappings such as *DisjointClass* and *OverlapClass*.
- Develop and publish reference ontologies explicitly defining common relations between concepts and properties, which remain neglected in existing ontologies, including disjointness relations and translation rules between common modelling paradigms.
- Maintain provenance and estimated reliability of automatically produced instance-level mappings so that an agent can make a decision about whether to use them or not.

As the top priorities for the future work currently we are considering the following:

- Continue more experimental testing with public linked data sources using detailed ontologies (such as DBPedia 3.2).
- Develop a data fusion service, which can operate on the Semantic Web in conjunction with existing linked data sources and semantic applications (such as WATSON, SCARLET, Alignment Server).

8. REFERENCES

[1] B. Aleman-Meza, C. Halaschek, A. Sheth, I. B. Arpinar, and G. Sannapareddy. SWETO: Large-scale Semantic Web test-bed. In *Workshop on Ontology in Action, 16th International Conference on Software Engineering and Knowledge Engineering (SEKE2004)*, pages 21–24, 2004.

[2] P. Bouquet, H. Stoermer, and B. Bazzanella. An Entity Name System (ENS) for the Semantic Web. In *5th Annual European Semantic Web Conference (ESWC 2008)*, pages 258–272, 2008.

[3] P. Bouquet, H. Stoermer, and D. Giacomuzzi. OKKAM: Enabling a web of entities. In *WWW2007 Workshop i3: Identity, Identifiers and Identification*, Banff, Canada, 2007.

[4] J. Euzenat. An API for ontology alignment. In *3rd International Semantic Web Conference*, volume 3298 of *Lecture Notes in Computer Science*, pages 698–712, Hiroshima, Japan, 2004. Springer.

[5] J. Euzenat and P. Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg, 2007.

[6] A. Ferrara, D. Lorusso, and S. Montanelli. Automatic identity recognition in the Semantic Web. In *Workshop on Identity and Reference on the Semantic Web, ESWC 2008*, Tenerife, Spain, 2008.

[7] H. Glaser, I. Millard, A. Jaffri, T. Lewy, and B. Dowling. On coreference and the Semantic Web. In *7th International Semantic Web Conference (ISWC 2008) (submitted)*, Karlsruhe, Germany, 2008.

[8] J. Gracia and E. Mena. Matching with CIDER: Evaluation report for the OAEI 2008. In *3rd Ontology Matching Workshop (OM'08) at the 7th International Semantic Web Conference (ISWC'08)*, Karlsruhe, Germany, 2008.

[9] R. V. Guha and R. McCool. TAP: a Semantic Web platform. *Computer Networks*, 42(5):557–577, 2003.

[10] A. Jaffri, H. Glaser, and I. Millard. Managing URI synonymity to enable consistent reference on the Semantic Web. In *Workshop on Identity and Reference on the Semantic Web (IRSW2008)*, Tenerife, Spain, 2008.

[11] M. Klein. Combining and relating ontologies: an analysis of problems and solutions. In *Workshop on Ontologies and Information Sharing*, 2001.

[12] Y. Liu, F. Scharffe, and C. Zhou. Towards practical rdf datasets fusion. In *Workshop on Data Integration through Semantic Technology (DIST2008), ASWC 2008*, Bangkok, Thailand, 2008.

[13] S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: A versatile graph matching algorithm. In *18th International Conference on Data Engineering (ICDE)*, pages 117–128, San Jose (CA US), 2002.

[14] A. Nikolov, V. Uren, E. Motta, and A. de Roeck. Integration of semantically annotated data by the KnoFuss architecture. In *16th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2008)*, Acitrezza, Italy, 2008.

[15] M. Sabou, M. d'Aquin, and E. Motta. Exploring the Semantic Web as background knowledge for ontology matching. *Journal of Data Semantics*, 2008.

[16] F. Scharffe and D. Fensel. Correspondence patterns for ontology alignment. In *16th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2008)*, pages 83–92, Acitrezza, Italy, 2008.

[17] G. Tummarello, R. Delbru, and E. Oren. Sindice.com: Weaving the open linked data. In *6th International Semantic Web Conference (ISWC/ASWC 2007)*, pages 552–565, 2007.

[18] P. Wang and B. Xu. Lily: Ontology alignment results for OAEI 2008. In *3rd Ontology Matching Workshop (OM'08) at the 7th International Semantic Web Conference (ISWC'08)*, Karlsruhe, Germany, 2008.