Towards instance coreference resolution in a multi-ontology environment

Andriy Nikolov[1], **Victoria Uren**[1], **Enrico Motta**[1] **and Anne de Roeck**[1]

1 INTRODUCTION

With the growing amount of semantic data published on the Web the problem of coreference resolution gains in importance. The linked data initiative provided guidelines for publishing RDF datasets and new datasets are constantly being made available. Such datasets often contain descriptions of the same real-world entities but use different URIs to refer to them. In order to utilize published data on a web scale it is essential to detect such situations and resolve coreferences.

In the Semantic Web community initially research effort was primarily concentrated on schema-level ontology alignment and many tools have been developed [1]. With the growing amount of published data instance-level integration issues also started to receive attention recently [2], [4]. These systems abstract from schema-level issues and focus on finding coreferent instances assuming their type and structure to be the same.

In our view there is still a gap concerning the study of the complete data integration workflow. On the one hand, schema alignment algorithms do not support the level of granularity necessary for data processing (e.g., applying different settings for individuals of different class). On the other hand, data-level integration tools assume schema-level issues to be resolved and do not consider implications of automated schema alignment. Our system KnoFuss was initially developed to perform integration of automatically extracted annotations structured according to a single common ontology. We extended it to operate in a multi-ontology environment and to utilise schema alignments produced by automatic ontology matching tools. Here we describe the resulting system workflow and first findings obtained during initial tests.

2 EXTENDING KNOFUSS ARCHITECTURE FOR MULTI-ONTOLOGY COREFERENCE RESOLUTION

KnoFuss architecture [6] implements a modular framework for semantic data fusion. The architecture focused on two main data fusion subtasks: coreference resolution (finding identical instances) and knowledge base updating (refining coreferencing results and resulting knowledge base taking into account ontological constraints and data conflicts). Algorithms performing fusion subtasks are represented as problem-solving methods [5]. Their capabilities (range of applicability and reliability of output) are formally defined using the fusion ontology.

Obviously, the behaviour of each algorithm differs depending on the task to which it is applied: reliability of name matching using string similarity differs when individuals belong to a generic class *foaf:Person* or a specific class *sweto:Computer_Science_Researcher,* the same string metrics cannot be applied when comparing paper titles and person names because the order of words in the name can differ, etc. These differences are represented using *application contexts*: bridges between a specific domain and a method. For a coreference resolution method the context may define more precise reliability estimation, narrowed range of applicability and extended set of relevant properties.

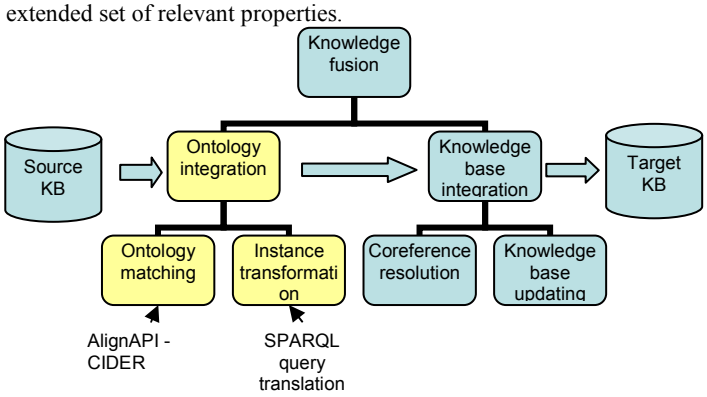

Figure 1. Fusion task decomposition in the KnoFuss architecture

Our ongoing work focuses on extending the functionality of KnoFuss to operate on a larger scale in a multi-ontology environment. If the source and target knowledge bases are structured according to different ontologies two additional subtasks are added to bridge the gap: *ontology matching* (obtaining schema alignments) and *instance transformation* (resolving structural differences between instances in two knowledge bases).

At the first step, schema-related statements are separated and available ontology matching algorithms are called to produce alignments (at the moment it is assumed that they produce their output in the standard AlignAPI format). The system utilizes two types of mappings: *equivalence* and *disjointness*. After candidate equivalence mappings are produced the system tries to generate additional disjointness relations:

- by inferring them using disjointness in a single ontology and available equivalence mappings;
- by querying background knowledge: the Scarlet service [7] is called to check whether disjointness between terms was defined in any other existing ontology on the Web.

In case of a logical conflict (e.g., when two classes are considered disjoint but their subclasses are equivalent), the conflict is resolved based on the similarity measure of corresponding mappings: less reliable mappings are excluded.

These automatically generated mappings are then used to perform instance transformation. In KnoFuss, applicability range and relevant attribute selection sets are defined as SPARQL queries in the terms of the target ontology. These queries are translated into the terms of the source ontology using available mappings. In the case when a term in the target ontology

potentially corresponds to several terms in the source ontology their union is considered: e.g.,

```
SELECT ?uri WHERE {
        ?uri rdf:type sweto:Computer_Science_Researcher }
is translated into
SELECT ?uri WHERE {
        {?uri rdf:type tap:CMU_Person}
UNION
        {?uri rdf:type tap:Computer_Scientist}
UNION
        {?uri rdf:type tap:Medical_Scientist}}
```

In this example there is a modelling style difference between two ontologies: individuals, which are classified as computer scientists in the SWETO ontology, are classified into several classes in TAP based on different criteria: place of work (*CMU_Person*) for some individuals and main research area (*Computer_Scientist* and *Medical_Scientist*) for others. Some authors of medical expert systems were assigned to the class *Medical_Scientist*. The ontology matching tool correctly identified these overlaps and produced three candidate mappings for the class *sweto:Computer_Science_Researcher*.

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.

3 RESULTS AND DISCUSSION

We implemented a prototype of the system employing the CIDER tool [3] and performed initial tests trying to find coreferent individuals in two testbed knowledge bases: TAP and SWETO (Table 1). We applied different string metrics over individuals of several classes. Also we ran tests applying the CIDER ontology matching tool to measure instance similarity to compare its performance to standard string similarity metrics.

Some general initial findings are:

- As could be expected, errors during the schema matching stage are propagated and can potentially lead to significant distortions during instance coreferencing. For instance (rows 5 and 6), incorrect alignment of *tap:Country* to *sweto:Company* led to 30% precision drop (many companies have names derived from country names).
- Ontological constraints are extremely valuable in coreferencing task as a mean to repair such errors. Apart from the widely used *owl:FunctionalProperty* and *owl:InverseFunctionalProperty*, which allow non-ambiguous instance identification, *negative* evidence is also valuable for 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 in rows 5 and 6. Most ontologies do not define these explicitly, however, having a high-level reference ontology accessible on the Web where these constraints are specified would be a significant source of information.
- Label comparison cannot be considered sufficiently reliable evidence for coreference resolution. 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.
- Although semantic heterogeneity (different meaning attached to similar resources) is primarily a schema-level knowledge modelling issue, it can cause problems on the instance level as well. For instance, the TAP ontology contains a single individual “Coca-Cola” while SWETO contains several individuals describing Coca-Cola branches in different countries. Whether such instances should be considered equivalent depends on the context of the task.
- Since errors are inevitable in automatic coreferencing, provenance information must be stored together with produced coreference mappings so that the user application can decide whether to rely on them or not. One possible way is to extend the coreference bundles approach [2] to include for each URI the confidence of its inclusion into the set.
- It is hard to find a single matching algorithm to apply to all kinds of data: settings have to be optimised for a specific type of data rather than for a specific pair of ontologies as in schema matching. For instance, optimal thresholds for CIDER differed significantly depending on the class (rows 2, 7 and 11).

REFERENCES

[1] Euzenat, J. and Shvaiko, P. Ontology Matching. Springer, 2007.

[2] Glaser, H., Millard, I., Jaffri, A., Lewy, T. and Dowling, B. On Coreference and The Semantic Web. In: *7th International Semantic Web Conference*, October 26 - 30, Karlsruhe, Germany. 2008 (Submitted)

[3] Gracia, J. and Mena, E., Ontology Matching with CIDER: Evaluation Report for the OAEI 2008, Proc. of 3rd Ontology Matching Workshop (OM'08), at 7th International Semantic Web Conference (ISWC'08), Karlsruhe (Germany), CEUR-WS, ISSN-1613-0073, October 2008.

[4] Yanbin Liu and Francois Scharffe and Chunguang Zhou. Towards Practical RDF Datasets Fusion. Workshop on data integration through semantic technology (DIST2008), Bangkok, Thailand, December 2008.

[5] E. Motta. Reusable Components for Knowledge Modelling. IOS Press, 1999.

[6] A. Nikolov, V. Uren, E. Motta, A. de Roeck. Integration of semantically annotated data by the KnoFuss architecture. EKAW, Acitrezza, Italy, 2008.

[7] Sabou, M., d'Aquin, M., Motta, E., Exploring the Semantic Web as Background Knowledge for Ontology Matching. Journal of Data Semantics XI, 2008.

N	Class (SWETO)	Algorithm	Precision	Recall	F1	Threshold
1	Person	L2 Jaro-Winkler	0.29	0.92	0.44	1.0
2	Person (2000 subset)	CIDER	0.37	0.95	0.53	0.05
3	Computer Science Researcher	L2 Jaro-Winkler	0.62	0.93	0.75	0.99
4	Organization	Monge-Elkan	0.42	0.95	0.58	0.93
5	Company	Monge-Elkan	0.41	0.95	0.57	0.92
6	Company (manual schema alignment)	Monge-Elkan	0.74	0.95	0.83	0.93
7	Company (2000 subset)	CIDER	0.93	0.64	0.76	0.14
8	Company (2000 subset)	Jaro-Winkler	0.79	0.74	0.77	0.92
9	City	Monge-Elkan	0.92	0.98	0.95	0.92
10	City (2000 subset)	Jaro-Winkler	0.80	0.99	0.89	0.92
11	City (2000 subset)	CIDER	0.91	0.61	0.73	0.28

Table 1. Subset of initial test results for instance coreferencing (TAP vs SWETO)