ATOM: Automatic Target-driven Ontology Merging

Salvatore Raunich, Erhard Rahm

University of Leipzig, Leipzig, Germany
{raunich, rahm}@informatik.uni-leipzig.de

***Abstract*— The proliferation of ontologies and taxonomies in many domains increasingly demands the integration of multiple such ontologies to provide a unified view on them. We demonstrate a new automatic approach to merge large taxonomies such as product catalogs or web directories. Our approach is based on an equivalence matching between a source and target taxonomy to merge them. It is target-driven, i.e. it preserves the structure of the target taxonomy as much as possible. Further, we show how the approach can utilize additional relationships between source and target concepts to semantically improve the merge result.**

I. Introduction

Ontologies and taxonomies are increasingly used to semantically classify or annotate information in a lot of different application contexts. For example, in life sciences, ontologies are used to describe components and functions of organisms or objects such as genes or proteins; on the web, product catalogs of online shops, comparison portals or web directories use taxonomies to classify products or websites and to help users and applications finding relevant information. Many ontologies have been designed for the same domain in an independent way and there is a growing need to integrate or merge them with the goal to create a single ontology providing a unified view on the input ontologies while maintaining all information coming from them.

The ontology integration problem was investigated during the last years, but it is still a challenge if one wants to perform the integration in a largely automatic way. The related research problem of schema integration has been studied thoroughly for a long time [2] but most earlier approaches suffered from trying to solve the complex problems of matching and merging in a single approach. More recent work on schema integration builds on the research results on semi-automatic schema matching [9] and separate matching from merging. Hence, several algorithms have been proposed to merge schemas based on a pre-determined match mapping [3], [11], [6], [8]. Despite this simplification, several of these merge approaches are still not fully automatic but depend on manual intervention. Previous approaches on ontology merging [5], [4], [12] are also user-controlled and do not utilize the separation of matching and merging. While user-controlled approaches provide flexibility for determining the merge result, they require the involvement of expensive data integration experts and introduce a substantial manual effort especially for large ontologies.

The AUTOMATIC TARGET-DRIVEN ONTOLOGY MERGING (ATOM) system is an attempt at overcoming these limitations. It implements a new approach for taxonomy merging which generates a default solution in a fully automatic way that may interactively be adapted by users if needed. It uses a target-driven algorithm, i.e. it merges a source taxonomy into the target taxonomy. Such an asymmetric merge is highly relevant in practice and allows us to incrementally extend the target ontology by additional source ontologies. For example, the catalog of a new online shop may be merged into the catalog of a price comparison portal.

We discuss the ATOM base algorithm that takes as input two taxonomies and an equivalence matching between concepts and we also present an extended algorithm that can utilize additional relationships between the input taxonomies to semantically improve the merging result. The algorithms generate taxonomies that preserve all instances of the input taxonomies as well as the structure of the target taxonomy. Furthermore, they try to limit the semantic overlap in the merged taxonomy, by utilizing the input mapping and giving preference to the target taxonomy when the same concepts are differently organized in source and target.

II. Overview

The example in Figure 1 shows two taxonomies classifying automobiles in different ways. The source taxonomy classifies first by body style (sedan, wagon, etc.) and then by manufacturer (Audi, BMW, etc.) while the target taxonomy uses the opposite order. An equivalence mapping between source taxonomies is already given, represented as a set of equivalence correspondences between source and target elements. In this paper, we will consider only acyclic taxonomies with is-a relationships but multiple inheritance is supported, i.e. a concept can have multiple parents. In order to simplify the visualization and to keep our examples more readable, in the following we only refer to taxonomies with single inheritance but the algorithms proposed in this paper can deal with both single and multiple inheritance. Furthermore, we assume that instances are limited to leaf nodes which is frequently the case in real applications. We established an extension of our algorithm to support instances for inner concepts but we cannot provide the details here due to space constraints. Each concept in a taxonomy can have attributes that will be merged if covered by one or more correspondences.

The motivation is that a taxonomy merging algorithm should merge equivalent concepts into a common concept and then correctly place the remaining source concepts in the merged taxonomy. In contrast to previous approaches we do not try to preserve all non-matching concepts and their relationships in both input ontologies since this could introduce a semantic

overlap in the merge result due to a different and overlapping structuring of the domain of interest. To limit this semantic overlap (redundancy), we focus on maintaining the target ontology and preserving all instances of both input ontologies while retaining only those non-matched source ontology concepts and relationships that are needed to preserve all instances.

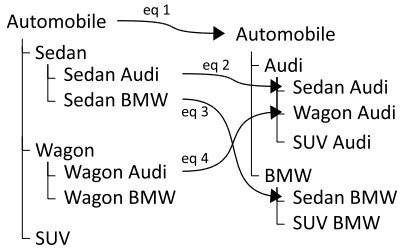

Fig. 1: Running Example

Properties of a merged taxonomy. Based on [7], we identified, adapted and extended some properties that the solution of our merge algorithm should satisfy. We give an informal description here, but a formal discussion can be found in [10].

Target element preservation (P1) and *Target relationship preservation (P2)* properties express how the merge result between a source and target ontology should be a taxonomy having a similar base structure than the target. It means that each target concept must be present in the merged taxonomy and all target is-a relationships between concepts must be semantically preserved. In addition to target maintenance, we require that all instances of both the target and the source ontology must be preserved in the merged taxonomy. We call this property *Information Preservation (P3)*. The *Control of semantic overlap (P4)* property suggests that the merge algorithm should also generate an integrated taxonomy with little or no redundancy compared to the input taxonomies. In particular, no instance overlap between concepts should be introduced. The last property, *Equality preservation (P5)*, states that if two concepts are equal in the equivalence mapping then they are mapped to the same merged concept in the result and vice versa.

Finally, we require that the algorithm must terminate and produce a result that is itself a taxonomy (respectively *Termination* and *Closure*) and should be *scalable* and able to provide good performance and acceptable execution times also for large taxonomies with many concepts and is-a relationships.

Base algorithm. We propose a base algorithm that, given as input two taxonomies and an equivalence mapping, produces an integrated taxonomy that satisfies properties discussed above. We can identify two successive phases in the algorithm: a *preliminary phase* and a *main phase*. More details of the algorithm can be found in [10].

The preliminary phase is responsible for creating an integrated concept graph I starting from two input taxonomies and an equivalence matching between them. An integrated concept graph contains all concepts coming from input taxonomies and the concepts mapped by the equivalence mapping are merged in one concept. For each is-a relationship in the input taxonomies, a labeled edge will be generated in the graph distinguishing source and target edges, called in the following *s-edges* and *t-edges*. A similar algorithm was introduced in [3] for relational and XML schemas and the preliminary phase of our approach is based on it but with significant differences. One of the main differences is the distinction in the integrated concept graph between edges coming from source and that ones coming from target; this distinction will play a very important role in the automatic generation of the final result. The integrated concept graph I generated for our running example is in Fig. 2. Nodes with a star (*) in the label represent merged concepts (e.g. *Sedan BMW** or *Wagon Audi**); edges with labels S_1 to S_7 are source edges and edges with labels T_1 to T_7 are target edges.

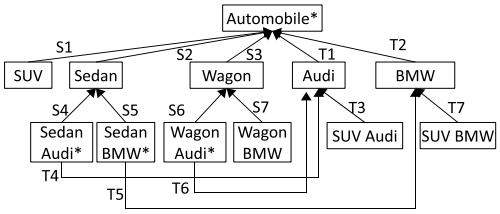

Fig. 2: Integrated Concept Graph

The main phase is based on a integrated graph visiting algorithm. The main idea is how source and target edges are translated. As first step we check and remove cycles in the graph I. Since we are assuming that input taxonomies are acyclic, any cycle in I cannot involve only s-edges or t-edges. It is worth to note that there can be more than one way to solve this kind of cycles. For example, in [7] all concepts involved in a cycle are merged in a single concept since is-a relationships are transitive and a similar cycle implies equality of all its concepts. In our algorithm we solve cycles in a different way, deleting one of the s-edges involved in the cycle. The intuition behind this choice is that we define our algorithm as target-driven in order to preserve the target structure in the final result, and the removal of a s-edge does not modify the target structure. In this step, the user might choose which edge to remove for producing a better solution.

In the second step we translate all t-edges. For each t-edge $e = N_1 \rightarrow N_2$, we normally create an is-a relationship between the integrated concepts C_1 and C_2 corresponding to N_1 and N_2 in I, respectively, in order to maintain the target concepts and relationships. The only exception is when there exists exactly one source path P with start node N_1 and end node N_2 containing more than one s-edge. In the latter case we do not create a direct relationship between C_1 and C_2 but we mark all edges in P as relevant (for the merged taxonomy). Thus, we want to preserve the target structure in the final result but if two concepts have a more detailed structure in the source, we want to reward it in the merged taxonomy since it preserves and extends the target structuring between N_1 and N_2.

The most important step in the main phase is the translation of s-edges because the algorithm tries to integrate in a correct

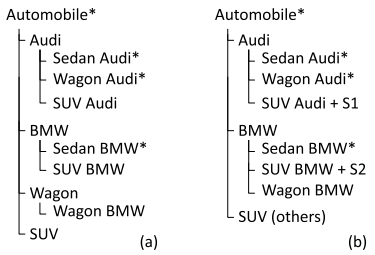

Fig. 3: Result of the base (a) and the extended (b) algorithms

place the missing concepts coming from the source taxonomy. The idea is that we want to check which s-edges are relevant for the merge result without introducing redundancy in addition to the t-edges. In order to do this, for each leaf node in the source, we traverse each path P from the node to the root containing only s-edges (that we call *s-path*) and consider its edges as relevant until one node in P has outgoing t-edges indicating that the remaining path is already covered by T.

In our running example, the set of source leaf nodes is {*Sedan Audi***, Sedan BMW***, Wagon Audi***, Wagon BMW, SUV*}; for example, the node *Wagon BMW* has the following s-path $P = \{S_7, S_3\}$. The start node of S_7, that is the first edge in P, is *Wagon BMW* and since it has no outgoing t-edges we mark S_7 as relevant; the same goes for S_3, the next edge in P, and we mark it as relevant. Instead, if we consider the node *Sedan Audi**, it has the following s-path $P = \{S_4, S_2\}$. The start node of S_4, the first edge in P, is *Sedan Audi** and since it has an outgoing t-edge, T_4, we do not mark S_4 as relevant because a translation for *Sedan Audi** has already been proposed in the previous step. At the end of this step, only S_1, S_3 and S_7 will be marked as relevant; it follows that *SUV*, *Wagon BMW* and *Wagon* will be translated in the merge result while *Sedan* is considered not relevant for the merged taxonomy and it will be ignored. Figure 3(a) shows the integrated taxonomy generated by the algorithm for our running example.

It is easy to see that the proposed algorithm meets the requirements (P1) to (P5) introduced in this section. By merging corresponding concepts we reduce semantic overlap compared to a simple union of the input taxonomies; furthermore we eliminate redundant inner nodes such as *Sedan* for the running example. Still there is remaining semantic overlap in the merge result determined by the base algorithm and we discuss next how to further reduce it with the extended algorithm. The default result generated automatically by the system might not satisfy the subjectivity of the user. In this case, ATOM shows which concepts were considered as not relevant and the list of the target concepts with their paths that make them redundant in order to help the user to choose a different solution.

Extended algorithm. Inspecting the result of the base algorithm, reveals that not all concepts seem well placed and that there is still some semantic overlap due to the differences in the original taxonomies. For example the concept *Wagon BMW* is not in the same subtree than concept *BMW*. Moreover, there is likely overlap between the general *SUV* concept under *Automobile** and the more specific concepts *SUV Audi* and *SUV BMW*. Since the provided equivalence mapping does not express these semantic relationships, we provide an extended algorithm using enriched input mappings with *is-a* and *inverse-isa* relationships. An is-a correspondence is an oriented correspondence from a source concept to a target concept and it expresses an is-a relationship between them; similarly we can define a inverse-isa correspondence. For example, we can specify an is-a correspondence to describe that the source concept *Wagon BMW* is a subclass of the target concept *BMW* and a pair of inverse-isa correspondences to state that the source concept *SUV* semantically is a superclass for both target concepts *SUV Audi* and *SUV BMW*, since *SUV* in the source represents every kind of SUV and not a SUV of a specific manufacturer, as instead in the target is. In order to utilize the semantic information coming from is-a and inverse-isa mappings, we introduce is-a edges and inv-isa-edges in the integrated concept graph reflecting respectively is-a and inverse-isa correspondences. The result generated by the extended algorithm for our running example is shown in Fig. 3(b). As we can see, the concept *Wagon BMW* is now correctly placed and the concept *Wagon* was ignored in the final result since considered as no more relevant. Moreover, concepts *SUV Audi* and *SUV BMW* were renamed respectively as *SUV Audi* + S_1 and *SUV BMW* + S_2 to indicate that they represent not only the target concepts but also a subset of the source concept *SUV*, as defined in the inverse-isa correspondences; finally the concept *SUV (others)* represents all source *SUV* not already merged into other concepts – informally speaking, it represents all SUV with a manufacturer different from *Audi* and *BMW*. During the demonstration we will discuss complex scenarios and we will show how the specification of is-a and inverse-isa relationships improves the quality of the merged result.

Mapping Generation. After the integrated taxonomy has been determined, equivalence mappings between the input taxonomies and the merged taxonomy can be automatically generated. The process is fully automatic and based on the extended integrated concept graph reflecting the equivalence, is-a and inverse-isa relationships between the input taxonomies. The idea is that these relationships produce different edges in the integrated concept graph and consequently different concepts and relationships in the merged taxonomy. In particular, correspondences in an equivalence mapping describe how two or more source concepts should be merged in the integrated taxonomy; on the other side, an isa-mapping does not produce merged concepts in the result, but it defines a subclass relationship between a source and a target concept, describing which should be the father of a source concept in the merged taxonomy; finally, an inverse-isa mapping describes how to split a source concept (and its instances) in two or more concepts in the final result. The algorithm will produce a correspondence for every leaf concept specifying where instances should migrate in the merged taxonomy. Fig. 4 shows the mappings M_1 and M_2 generated for our running example, for relating the input taxonomies to the integrated taxonomy; we have drawn correspondences between leaf nodes with a

solid line and that ones between inner nodes with a dotted line. Thus, properties (P3) and (P4) are satisfied with respect to instances, since there is a correspondence for each leaf node in source taxonomies and each instance migrates to exactly one concept in the merged taxonomy.

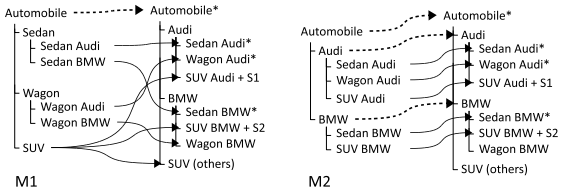

Fig. 4: Mappings M_1 and M_2

III. Demonstration

ATOM has been implemented as a working prototype written in Java, offering a GUI to explore all steps of the merging generation process. During the demonstration we will show various usage scenarios supported by the system. In a typical scenario, a user chooses a merging scenario providing input taxonomies and mappings; correspondences can be loaded with input taxonomies or they can be manually drawn using the GUI as shown in Figure 5. ATOM has been integrated with COMA++ [1] which permits semi-automatic generation of input mappings. The user can specify different kinds of mappings: equivalence, is-a and inverse-isa correspondences are supported by the system as discussed in the previous section, so that the user can decide if the merge solution must be generated using the base or the extended merge algorithm. At the end of the merge process, the system shows the integrated taxonomy highlighting which concepts have been merged, which ones have been generated from is-a mapping and which ones from inverse-isa mapping. The set of source nodes marked as not relevant by the algorithm and ignored in the final result can be inspected by the user. Finally, the system automatically generates equivalence mappings between the input taxonomies and the merged taxonomy and it shows them to the user.

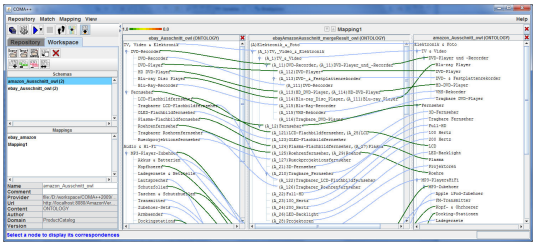

Fig. 5: A snapshot of the system

The demonstration will focus on the discussion of various merging scenarios, showing the features and the quality of the merge result produced by ATOM . To show how the solution produced by the system satisfies the properties discussed in Section II, we have prepared synthetic scenarios of small and medium sizes in order to easily inspect the final result. These scenarios will be also used to show how the specification of a more semantic mapping between the input taxonomies, in particular is-a and inverse-isa relationships in addition to equivalence relationships, can reduce the semantic overlap in the merge taxonomy.

We will show how in practical cases ATOM computes the merge taxonomy very efficiently and scales well to large taxonomies in various real-life scenarios. We distinguish two main scenarios: the *Anatomy (Mouse-NCI)* scenario and the eBay product catalog scenarios. The first one merges the AdultMouseAnatomy (over 2700 concepts) with the anatomical part of the NCI Thesaurus (NCIT) (about 3300 concepts); the second one merges different versions of the eBay product catalog with over 20000 concepts. The input taxonomies in the Anatomy scenario have a limited overlap since only 33% input concepts are mapped by an equivalence correspondence. This scenario shows that only few source concepts are considered as not relevant for the merged taxonomy and were ignored. The system scaled well to this scenario since it produced the merged result in about one second. The eBay product catalog scenarios will be useful to show the scalability of the system to very large ontologies. We considered the same scenario where the input taxonomies are two successive versions of the eBay product catalog. The system scaled well also on these scenarios (the maximum execution time took less than 10 seconds) and during the demonstration we will discuss how the proposed solution reduces the semantic overlap.

IV. Conclusions

With the ATOM system, we present a novel merging tool that combines taxonomies by integrating one into another one. We demonstrate the successfulness of our approach by applying it on large real life ontologies.

References

[1] D. Aumueller, H. Do, M. S., and E. Rahm, "Schema and Ontology Matching with COMA++," in *Proc. of ACM SIGMOD*, 2005, pp. 906–908.

[2] C. Batini, M. Lenzerini, and N. S. B., "A Comparative Analysis of Methodologies for Database Schema Integration," *ACMCompSurv*, vol. 18, no. 4, pp. 323–364, 1986.

[3] L. Chiticariu, P. G. Kolaitis, and L. Popa, "Interactive generation of integrated schemas," in *Proc. of ACM SIGMOD*, 2008, pp. 833–846.

[4] D. L. McGuinness, R. Fikes, J. Rice, and S. Wilder, "An environment for merging and testing large ontologies," in *KR*, 2000, pp. 483–493.

[5] N. F. Noy and M. A. Musen, "Prompt: Algorithm and tool for automated ontology merging and alignment," in *AAAI/IAAI*, 2000, pp. 450–455.

[6] R. Pottinger and P. A. Bernstein, "Schema merging and mapping creation for relational sources," in *EDBT*, 2008, pp. 73–84.

[7] R. A. Pottinger and P. A. Bernstein, "Merging models based on given correspondences," in *Proceedings of the 29th International Conference on Very Large Data Bases - Volume 29*. VLDB Endowment, 2003, pp. 862–873.

[8] A. Radwan, L. Popa, I. R. Stanoi, and A. A. Younis, "Top-k generation of integrated schemas based on directed and weighted correspondences," in *SIGMOD Conference*, 2009, pp. 641–654.

[9] E. Rahm and P. A. Bernstein, "A Survey of Approaches to Automatic Schema Matching," *VLDB J.*, vol. 10, pp. 334–350, 2001.

[10] S. Raunich and E. Rahm, "Target-driven Merging of Taxonomies," University of Leipzig, Tech. Rep., 2010. [Online]. Available: http://arxiv.org

[11] K. Saleem, Z. Bellahsene, and E. Hunt, "Porsche: Performance oriented schema mediation," *Inf. Syst.*, vol. 33, no. 7-8, pp. 637–657, 2008.

[12] G. Stumme and A. Maedche, "Fca-merge: Bottom-up merging of ontologies," in *IJCAI*, 2001, pp. 225–234.