Toward OWL Restriction Reconciliation in Merging Knowledge

Elena Grygorova* [1], Samira Babalou [1], Birgitta König-Ries [1 2]

[1]Heinz-Nixdorf Chair for Distributed Information Systems
Institute for Computer Science, Friedrich Schiller University Jena, Germany
[2]Michael-Stifel-Center for Data-Driven and Simulation Science, Jena, Germany
{elena.grygorova,samira.babalou,birgitta.koenig-ries}@uni-jena.de

Abstract. Merging ontologies is the standard way to achieve interoperability of heterogeneous systems in the Semantic Web. Because of the possibility of different modeling, OWL restrictions from one ontology may not necessarily be compatible with those from other ontologies. Thus, the merged ontology can suffer from restriction conflicts. This problem so far has got little attention. We propose a workflow to detect and resolve the OWL restriction conflicts within the merged ontology. We reconcile "one type" conflicts by building a subsumption hierarchy. We tackle cardinality restriction conflicts with least upper and greatest lower bound methods. By utilizing the semantic relatedness between two classes, we overcome value restriction conflicts.

Keywords: Semantic Web . Ontology Merging . OWL Restriction Conflict

1 Introduction and Related Work

Ontology merging [1] is the process of creating a merged ontology $\mathcal{O}_M$ from a set of source ontologies $\mathcal{O}_S$ based on given mappings. In ontologies, OWL classes are described through class expressions to represent real-world constraints, such as type, cardinality or value restrictions. However, two ontology developers may model the same or overlapping entities to describe the common real-world objects with different restrictions. When two different restrictions are combined in the merged ontology, conflict can happen easily. Finding a compromise between restrictions in the merged ontology is introduced as one of the Generic Merge Requirements (GMR)s in [2]. Representing conflicts has been considered as a significant challenge for integration methodologies for a while. Existing approaches address either data-level conflicts [3], or schema-level conflicts [4], or structural conflicts [5], only.

We develop a workflow that detects and resolves OWL restriction conflicts. We build a subsumption hierarchy over datatypes to reconcile "one type" conflicts. To detect and resolve OWL cardinality and value restriction conflicts, we build an attribute restriction graph. Cardinality restriction conflicts are tackled with least upper and greatest lower bound methods. By utilizing the semantic relatedness between two classes, we overcome value restriction conflicts.

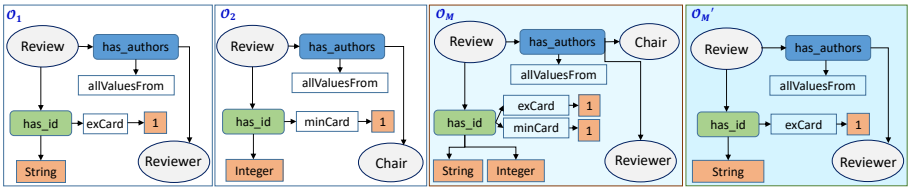

Fig. 1: Fragments of $\mathcal{O}_1$ and $\mathcal{O}_2$, the conflicting $\mathcal{O}_M$, and the repaired $\mathcal{O}'_M$.

2 Proposed Reconciliation Method

Datatype and object properties in an ontology represent the context and the semantics of concepts. They obey a set of restriction rules. Putting them together in the merged ontology can result in either a *compatible* or a *conflicting* merged ontology with the following definition:

Definition 1. *The merged ontology is* ***compatible*** *if no conflicts exist. If there is at least one conflict over its restrictions, we have a* ***conflicting*** *merged ontology.*

Conflicting merged ontologiy contains a set of restriction conflicts: (i) "one type" conflicts, (ii) value and cardinality restriction conflicts.

(i) "One type" Conflicts: Detection & Solution. A datatype property should have at most one range. This has been called the one-type restriction [1]. A conflict can happen in the merged ontology when two corresponding entities from different source ontologies have different data types[1]. For example, in the ontology fragments in Fig. 1, `has_id` from $\mathcal{O}_1$ and $\mathcal{O}_2$ contain two different datatypes: `String` and `Integer`. In the merged result $\mathcal{O}_M$, the two corresponding `has_id` are integrated into one entity. However, the type entities remain separate, so `has_id` is the origin of two type relationships, which indicates a "one type" conflict.

The first step toward reconciling "one type" conflicts is to determine which alternative data type can be used in the merged ontology. We build a *Subsumption Hierarchy* $\mathcal{SH}$ over all supported datatypes in OWL Full. The subsumption relations between the datatypes in $\mathcal{SH}$ are built based on the general data types conversions[2]. Starting from depth zero at the root, the most general datatype comes on the next level. After that, more precise datatypes are considered. The depth of each datatype $depth(v_i)$ shows its level in the $\mathcal{SH}$. For example, the depth of `Float` in $\mathcal{SH}$ is less than the depth of `Double` because `Double` is more precise than `Float`.

Definition 2. *Two data types are* ***compatible*** *if there is a path in* $\mathcal{SH}$ *between them that does not go through the root. Otherwise, they are* ***incompatible****.*

[1] The one type conflict can happen only on datatype properties.

[2] We assumed the OWL/RDF data types could be mapped to Java data types and considered the general data types conversions from: https://docs.oracle.com/javase/specs/jls/se7/html/jls-5.html#jls-5.5

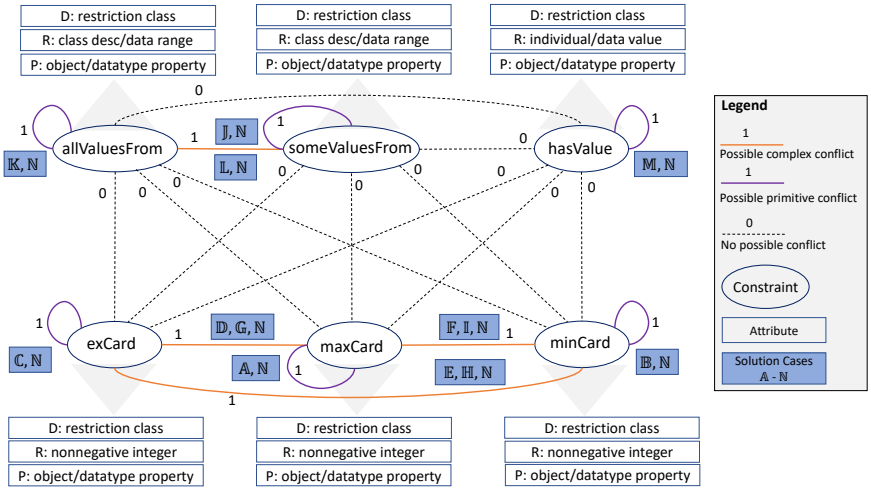

Fig. 2: The attributed Restriction Graph $\mathcal{RG}$ for six OWL Restriction types, showing the interaction and solution cases.

Substitution *of two compatible data types* $v_i, v_j \in \mathcal{SH}$ *with* $depth(v_i) < depth(v_j)$*, is the type of* v_i*, since* v_i *is a more general type in* $\mathcal{SH}$*.*

If v_i and v_j are compatible and have the same depth, e.g., are siblings, the substitution is the parent type of both in $\mathcal{SH}$. If v_i and v_j are incompatible, then no substitution can be performed on them. In this case, we follow the proposed solution in the early work of the schema merging aspects in [6]. This resolution creates a completely new type that inherits from both data types and replaces the two type-of relationships from the respective property by one type-of relationship to the new type. Thus, for two contradicting values, an **instantiation** of them is a new inherited type of both. The proposed approach is valid when the values of the restrictions are data types, i.e., `String`, `Integer`, `Float`. If they are class types (e.g., Man, Woman, Person), we follow the semantic relatedness strategy, which we will discuss in the next part.

(ii) Value and Cardinality Restriction Conflicts: Detection & Solution. An ontology may restrict the maximum or a minimum number of occurrences that a given entity can take part in a relationship or enumerate the possible values of properties. However, when the source ontologies place restrictions on a property's values, the merged ontology may exhibit conflicts [2]. To detect and reconcile value and cardinality restriction conflicts, we build an attributed Restriction Graph $\mathcal{RG}$ for the six OWL restrictions (see Fig. 2). A $\mathcal{RG} = (V, E)$ is an undirected labeled graph, where V is a set of vertices, and E is a set of edges. The vertices correspond to the values and cardinality restrictions, while the edges show the interactions between the vertices. Each vertex holds three attributes: Domain (D), Range (R), and the properties (P) on which the

constraint can be applied. A constraint links a Domain to a Range and can be applied on object or datatype properties. Domain (D) and Properties (P) attributes for our vertices have the same values. Thus, we construct the edges based on the Range (R) attribute, as given by Def. 3 and 4. The interactions between vertices can reveal three different states: (1) no conflict (isolated), (2) primitive, or (3) complex conflict.

Definition 3. *If the Range (R) attributes of two vertices in the $\mathcal{RG}$ are the same, depending on their values and ranges, there is a possibility of* ***conflict*** *for them. However, two restrictions with different Range (R) attributes are* ***isolated*** *from each other and can not have any conflicts.*

When there is a possible conflict between vertices, the edge between these two vertices holds label 1. Otherwise, labels of the edges are 0.

Definition 4. *A* ***primitive*** *conflict is a possible conflict between the same restriction types. A possible conflict over different restriction types is called a* ***complex*** *conflict.*

In the $\mathcal{RG}$ depicted in Fig. 2, all recursive violet-colored edges are types of possible primitive conflicts. Orange edges between two vertices in $\mathcal{RG}$ depict possible complex conflicts. Each primitive or complex conflict on the values or cardinality constraints requires a reconciliation method. We developed such methods and derived a detailed solution[3] to all 21 interaction restriction cases given by the cases $\mathbb{A}$-$\mathbb{N}$ in Fig. 2. A summary of the resolution is:

- **Cardinality restriction conflicts solution**: We use greatest lower and least upper bound methods adapted to the individual cases.
- **Value restriction conflicts solution**: When the value restriction is on a data property, we follow the approach described in Sec. 2-(i). If the value restriction is related to an object property, we apply the *semantic relatedness* solution, in this way, if two values are semantically related, following the generalization of them, we choose the super class out of them. If the values are siblings, we select the parent value of them. When there is no semantic relatedness for two values (i.e., they are not on the same hierarchy), no automatic reconciliation can be made.

3 Use Case Study

We have provided a preliminary evaluation of our proposed method. To this end, we conducted an experimental test on three pairs of ontologies adapted from the *conference* domain of the OAEI benchmark[4] provided by the OntoFarm project [7]. We observed how easily the small ontologies can cause conflicts when being merged, as they are augmented with many properties and constraints. We

[3] https://github.com/fusion-jena/CoMerger/blob/master/Restriction/solution.md
[4] http://oaei.ontologymatching.org/2019/conference/index.html

apply our strategy to solve existing conflicts. We then compare the conflicting merged ontology with the revised one with a set of Competency Questions[5]. The merged ontology that was revised by our approach could achieve homogenous answers, whereas the conflicting one returns contradicting answers. This test demonstrates that applying our method on the conflicting merged ontology can provide homogenous answers and shows the applicability of our method in practice.

4 Conclusion

Differences in modeling common entities can cause different types of conflicts when merging ontologies. In this paper, we tackled (i) "one type" conflicts by building a subsumption hierarchy on data types and performing substitution or instantiation on them, (ii) cardinality restriction conflicts with least upper and greatest lower bound method, (iii) value restriction conflicts by utilizing the semantic relatedness. A preliminary evaluation on a use case study shows the feasibility of our method. We plan to extend our experiments on the large scale ontologies in different domains such as biomedicine. Analyzing the effect of caused conflict on the instance level is on our future agenda.

Acknowledgement

S. Babalou is supported by a scholarship from German Academic Exchange Service (DAAD).

References

1. R. A. Pottinger and P. A. Bernstein, "Merging models based on given correspondences," in *VLDB*, pp. 862–873, 2003.
2. S. Babalou and B. König-Ries, "GMRs: Reconciliation of generic merge requirements in ontology integration," *In SEMANTICS Poster and Demo.*, 2019.
3. S. Sonsilphong, N. Arch-int, S. Arch-int, and C. Pattarapongsin, "A semantic interoperability approach to health-care data: Resolving data-level conflicts," *Expert Systems*, vol. 33, no. 6, pp. 531–547, 2016.
4. M. d. C. M. Batista and A. C. Salgado, "Information quality measurement in data integration schemas.," in *QDB*, pp. 61–72, 2007.
5. M. Fahad, "Merging of axiomatic definitions of concepts in the complex owl ontologies," *Artificial Intelligence Review*, vol. 47, no. 2, pp. 181–215, 2017.
6. P. Buneman, S. Davidson, and A. Kosky, "Theoretical aspects of schema merging," in *EDBT*, pp. 152–167, Springer, 1992.
7. O. Zamazal and V. Svátek, "The ten-year ontofarm and its fertilization within the onto-sphere," *Journal of Web Semantics*, vol. 43, pp. 46–53, 2017.

[5] https://github.com/fusion-jena/CoMerger/blob/master/Restriction/caseStudy.md