$\mathcal{CoMerger}$: A Customizable Online Tool for Building a Consistent Quality-Assured Merged Ontology

Samira Babalou* [1], Elena Grygorova [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
{samira.babalou,elena.grygorova,birgitta.koenig-ries}@uni-jena.de

Abstract. Merging ontologies enables the reusability and interoperability of existing knowledge. With growing numbers of relevant ontologies in any given domain, there is a strong need for an automatic, scalable multi-ontology merging tool. We introduce $\mathcal{CoMerger}$, which covers four key aspects of the ontology merging field: compatibility checking of the user-selected Generic Merge Requirements (GMR)s, merging multiple ontologies with adjustable GMRs, quality assessment of the merged ontology, and inconsistency handling of the result. $\mathcal{CoMerger}$ is freely accessible through a live portal and the source code is publicly distributed.

Keywords: Multiple ontology merging . Generic Merge Requirements . Ontology quality assessment . Ontology inconsistency

1 Introduction

Ontology merging is needed for many Semantic Web applications from a wide variety of domains. Therefore, there is a strong need for efficient, scalable, and customizable ontology merging tools. This has resulted in the development of several merging tools, including [1–6]. However, none of them meets all three requirements: methods in [2–6] are restricted to merging two ontologies at a time and are thus not sufficiently scalable. A set of pre-defined merge requirements is implemented in [5, 6] and thus they lack customization. Approaches in [1–5] lack the ability for users to assess the quality of the merged results and do not provide inconsistency handling. Lastly, to the best of our knowledge, none of them are available as web-based applications.

We propose $\mathcal{CoMerger}$ as a first step towards a comprehensive merging tool focussing on four important aspects: (i) compatibility checking of the user-selected Generic Merge Requirements (GMR)s [7], (ii) merging multiple ontologies with adjusting a set of user-selected GMRs [8], (iii) assessing the quality of the merged ontology [9], and (iv) inconsistency handling of the result [10]. This paper presents the architecture of $\mathcal{CoMerger}$ tool and the interaction between the mentioned aspects.

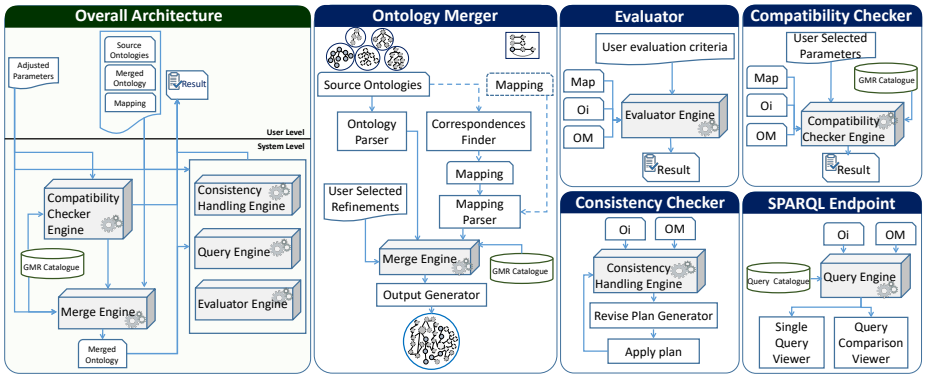

Fig. 1: $\mathcal{CoMerger}$ architecture and components.

2 Tool Overview

Fig. 1 shows the architecture of $\mathcal{CoMerger}$ components and the interaction between them in distinct boxes. The *Overall Architecture*, the left most box in Fig. 1 depicts the data flow between the $\mathcal{CoMerger}$ components, in two levels. The *user level* allows a user to interact with the tool through a friendly GUI. In the *system level*, the communication between the components is sketched. The user uploads a set of source ontologies alongside with the respective mappings[1]. If no mapping is given, $\mathcal{CoMerger}$ automatically detects the correspondences by using embedded ontology matching methods[2]. Moreover, the user is able to select from a set of twenty Generic Merge Requirements (GMR)s, including, e.g., entities preservation, one type restriction, acyclicity, and connectivity. The *Compatibility Checker* engine determines whether it is possible to simultaneously meet all requirements or there are contradictions. For instance, one may want to preserve all classes from the source ontologies in the merged ontology. On the other hand, one could wish to achieve class acyclicity. Likely, these goals conflict. The engine suggests a compatible subset of the GMRs given by the user. After parsing the source ontologies and their mappings, the merged ontology is automatically generated via the *Merge* engine by taking into account the user-selected GMRs.

Afterwards, the quality of the merged ontology can be evaluated via the *Evaluator* engine according to the user-selected evaluation aspects. Furthermore, there is a possibility to evaluate the quality of any given merged ontology independent of the merge process via a separate interface, *Evaluator*. Besides the quality criteria, the *Consistency Handling* engine can validate whether the merged result is consistent and provide support in repairing any issues.

[1] The tool can read a set of RDF alignment type, containing the similarity relations between entities with at least a given similarity value.

[2] Currently, two ontology matching approaches are embedded in our tool: SeeCOnt method [11] and a string matching based on the Jaccard similarity coefficient [12].

Additionally, through the embedded SPARQL endpoint of the *Query* engine, the user is able to compare query results on the merged and source ontologies. In the following, we describe the main components of Fig 1 in more detail.

2.1 Compatibility checker: A graph-based theory method

GMRs are a set of Generic Merge Requirements that the merged ontology is expected to achieve. The tool enables the flexible ontology merging process, in which the users can adjust a set of GMRs. However, not all GMRs are compatible. Thus, the compatibility checker component in $\mathcal{C}o\mathcal{M}erger$ verifies which GMRs can be met simultaneously. We utilized a graph-theory based method to capture the maximal compatible superset of user-selected GMRs. Our embedded twenty GMRs with the compatibility checker method have been presented in [7]. Up to now, it is not possible to extend the list of GMRs. Since this list covers all requirements towards merged ontologies mentioned in the literature, we believe that it is unlikely that the need will arise. Should that be the case, the tool could be adapted.

2.2 Ontology Merger: A partitioning-based approach

Our proposed merge method takes as input a set of source ontologies alongside the respective mappings and automatically generates a merged ontology. At first, the n ($n \geq 2$) source ontologies are divided into k ($k << n$) blocks and a local refinement is applied to them. After that, the blocks are combined to produce the merged ontology followed by a global refinement. The user can adjust a set of refinement operations via the embedded GMRs. Moreover, the tool logs the knowledge-level of the ontology merging process and the refinement operations, which can be further analyzed by the users. The whole underlying merge method is described in [8] and evaluated on various datasets. We compared the efficiency of our single step merged method with a series of pairwise merges. The results[3] demonstrate the high performance and quality of our method.

2.3 Evaluator: Quality assessment of the merged ontology

The merged ontology plays a central role in a variety of Semantic Web applications. Thus, prior to its usage, the quality and correctness of the merged ontology should be assessed. We provided a comprehensive set of evaluation criteria [9] to cover a variety of characteristics of each individual aspect of the merged ontology in three dimensions: (1) structural criteria via the evaluation of the General Merge Requirement (GMR)s, (2) functional measurements by the intended use and semantics of the merged ontology, and (3) usability-profiling evaluation on ontology and entity annotation. Our evaluation criteria also represent an analytic view on how well the created merged ontology reflects the given source ontologies. Evaluating the merged ontology can be performed even independently of the merge method by the separated interface in $\mathcal{C}o\mathcal{M}erger$.

[3] https://github.com/fusion-jena/CoMerger/blob/master/MergingDataset/result.md

2.4 Consistency checker: A Subjective Logic-based approach

The merged ontology should be free of any inconsistencies. However, since the encoded knowledge of source ontologies may model different world views, it can easily happen that the merged ontology is inconsistent. It needs to be resolved if one wants to make use of the merged ontology in further applications. Thus, we developed a Subjective Logic-based method in [10] to rank the conflicting axioms, which caused inconsistencies in the merged ontology. The rank function concerns the degree of trustworthiness of the source ontologies knowledge. Upon that, the tool suggests the remedies of changes such as deleting or rewriting a part of conflicting axioms to turn the inconsistent merged ontology into a consistent one. The whole process can be accomplished automatically, or a user can review the system's suggestions and make necessary changes before applying them.

3 Demonstration

In this demo[4], visitors will be able through our friendly GUI (see Fig. 2) specify requirements, ask for their compatibility, obtain suggestions for compatible subsets and a possible suggest compatible set, perform merge obeying the requirements, analyze the quality of the merged result w.r.t. their selected evaluation aspects, and check for consistency of the merged ontology. Users can save the merged ontology and the evaluation's results. For each selected evaluation criteria, they will receive the detailed result of the evaluation, as shown in Fig. 3. We will provide users with example ontologies, but they are also welcome to explore the tool with their own source ontologies. If interested, users can also directly access the source code, which is publicly available[5] and distributed under an open-source license. Our web-based application is supported by many modern web browsers. The host server (VM) for the tool includes 8 cores with CPU 2.39 GHz and 16 GB RAM. The processing time based on the size and number of source ontologies is reasonable. For instance, merging 17 ontologies with 51461 axioms took 140 seconds with a home internet (44 Mbps speed) in the Firefox 72.0.2 web browser. Users can opt for a local installation of the tool to omit delays due to network communication. We performed an experimental test on a local machine with Intel core i7 with 12 GB internal memory on Windows 7 with Java compiler 1.8. For 7, 22, 55 source ontologies with 3037, 56893, 158567 axioms, the merge method performs in 1.8, 62.3, 150.7 seconds, respectively. This demonstrates that the merge method in this tool scales well in the number and size of the source ontologies.

4 Related Work

Ontology merging has attracted considerable attention within the research community. Chiticariu et al. [1] proposed a method to enumerate multiple

[4] http://comerger.uni-jena.de/
[5] https://github.com/fusion-jena/CoMerger

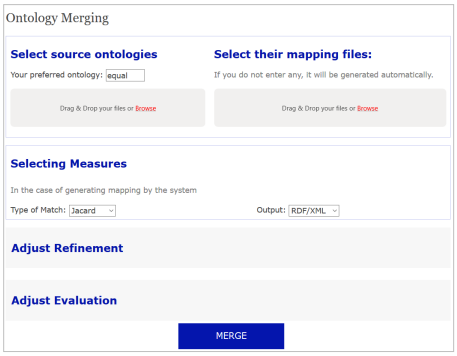

Fig. 2: Ontology merging GUI.

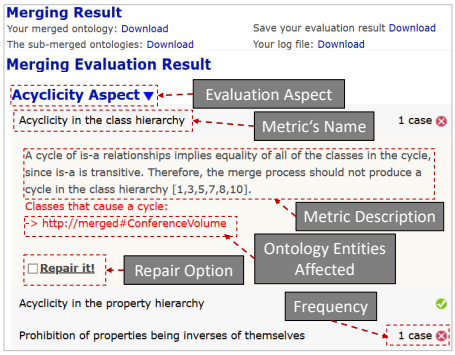

Fig. 3: Result of merge and evaluation.

integrated schemas from a set of source schemas by considering all possible choices of merging concepts. GROM [2] uses typed graph grammars with algebraic graph transformations. iPrompt [6] is an interactive ontology merging tool introduced as the Protégé-based implementation. This system leads users to perform merge by suggesting what should be merged. HSSM [3] generates the formal context for the source ontologies and merge the similar concepts within the built concept tree. GCBOM [4] applies the granular computing processes in order to produce the final merged ontology. ATOM [5], at first, creates an intermediate merged result, then refines it based on some of GMRs.

Despite the effort of many research studies, the developed ontology merging systems still suffer specific problems. In [1,6], many user interactions are required, which might not be feasible for large-scale ontologies. iPrompt [6] requires user interaction for all entity merging, and in [1], the enumerated schemas should be manually refined by users. To scale to many sources, the merging systems in [2–6] are insufficient due to merging only two ontologies at a time. No inconsistency handling is provided in [1–5]. In [5, 6], a set of fixed GMRs is implemented without user customization. To the best of our knowledge, besides iPrompt, the other mentioned systems are not publicly accessible and reproducible. Moreover, none of them are available as a web-based application.

5 Future Work

In our future work, we plan to extend $\mathcal{CoMerger}$ with respect to several dimensions: First, we will integrate the possibility to evaluate against a set of Competency Questions in the functional dimension to facilitate many use-case scenarios. Second, embedding other existing matchers in our tool and evaluating the source ontologies before the merge process might give a useful insight to the users. Finally, we plan a user study to evaluate the ease-of-use.

Acknowledgments

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

References

1. L. Chiticariu, P. G. Kolaitis, and L. Popa, "Interactive generation of integrated schemas," in *ACM SIGMOD*, pp. 833–846, 2008.
2. M. Mahfoudh, L. Thiry, G. Forestier, and M. Hassenforder, "Algebraic graph transformations for merging ontologies," in *MEDI*, pp. 154–168, Springer, 2014.
3. M. Priya and A. K. Ch, "A novel method for merging academic social network ontologies using formal concept analysis and hybrid semantic similarity measure," *Library Hi Tech*, 2019.
4. M. Priya and C. A. Kumar, "An approach to merge domain ontologies using granular computing," *Granular Computing*, pp. 1–26, 2019.
5. S. Raunich and E. Rahm, "Target-driven merging of taxonomies with atom," *Inf. Syst.*, vol. 42, pp. 1–14, 2014.
6. N. F. Noy and M. A. Musen, "The prompt suite: interactive tools for ontology merging and mapping," *IJHCS*, vol. 59, no. 6, pp. 983–1024, 2003.
7. S. Babalou and B. König-Ries, "GMRs: Reconciliation of generic merge requirements in ontology integration," *In SEMANTICS Poster and Demo.*, 2019.
8. S. Babalou and B. König-Ries, "Towards building knowledge by merging multiple ontologies with CoMerger: A partitioning-based approach," *http://arxiv.org/abs/2005.02659*.
9. S. Babalou, E. Grygorova, and B. König-Ries, "How good is this merged ontology?," in *In 17th Extended Semantic Web Conference (ESWC'20), Poster and Demo Track.*, June, 2020.
10. S. Babalou and B. König-Ries, "A subjective logic based approach to handling inconsistencies in ontology merging," in *OTM*, Springer, 2019.
11. A. Algergawy, S. Babalou, M. J. Kargar, and S. H. Davarpanah, "Seecont: A new seeding-based clustering approach for ontology matching," in *ADBIS*, pp. 245–258, 2015.
12. P. Jaccard, "Étude comparative de la distribution florale dans une portion des alpes et des jura," *Bull Soc Vaudoise Sci Nat*, vol. 37, pp. 547–579, 1901.