Partitioning-based Ontology Matching Approaches: A Comparative Analysis

Alsayed Algergawy[1,2], Friederike Klan[1], and Birgitta Konig-Ries[1]

[1] Institute for Computer Science, Friedrich Schiller University of Jena, Germany
[2] Department of Computer Engineering, Tanta University, Egypt
{firstname.lastname@uni-jena.de}

Generic Framework. Ontology matching is the process that takes two or more ontologies to identify semantically corresponding entities across them. As the numbers of developed ontologies as well as the number of entities in each ontology are increasing, traditional approaches to ontology matching fail or are not able to scale. Therefore, there is a growing need for new matching algorithms. A common approach to deal with the large-scale matching problem is the partitioning-based technique [5]. To make these techniques comparable, we propose a generic framework containing the following phases (shown in Fig. 1):

- *Prematch.* This phase aims to prepare input ontologies for matching. It starts by parsing and representing input ontologies as graphs, called *ontology graphs.* The input ontology graphs are then partitioned into a set of sub-ontologies such that entities belonging to one partition are similar (have some common features) while entities from different partitions are dissimilar. The partitioning process

Fig. 1: Partitioning-based matching steps.

may extend from using simple ad hoc rules [2] to clustering algorithms [1,4]. The task now is to determine which partitions of the two sets are sufficiently similar and thus worth to be matched in more detail. The goal is to reduce the matching overhead by avoiding to find correspondences between unrelated partitions.

- *Match.* Once settling on similar partitions (clusters) of the two ontologies, the next step is to fully match similar clusters to obtain the correspondences between their elements. Each pair of similar partitions represents an individual match task that is independently solved.

- *Postmatch.* Local match results should be merged (combined) to generate the final match result. The Postmatch phase is also concerned with *matching cardinality* and *mapping representation.*

Matching Systems: A Comparison. We aim to present partitioning-based approaches fitting to the algorithmic steps identified above indicating which part of the solution is covered by which prototypes, thereby supporting a comparison of these approaches. We notice that all these approaches use the graph
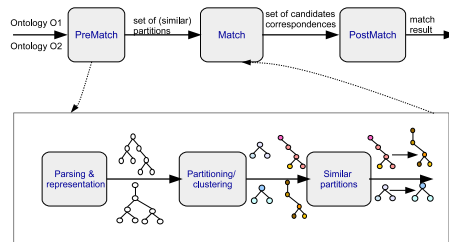
data structure as the internal data representation. However, they utilize different algorithms to partition the ontology graph. Falcon-AO [4] and the extension of COMA++ [1] employ an agglomerative clustering algorithm, which independently partitions input ontologies. To dependently partition ontologies, TaxoMap [3] uses a co-clustering technique. It is worth noting that some matching approaches first partition the ontology graphs and then determine similar partitions such as COMA++ [1,2] and Falcon-AO [4], while others determine similar partitions during the partitioning process such as TaxoMap [3]. We also observe that to determine similar partitions the matching approaches use different methods extending from exploiting only the partitions' roots, e.g. COMA++ [2], to exploiting the whole partition information, e.g. Falcon-AO [4]. Some other approaches compromise between the two extremes, e.g. the extension of COMA++ [1] exploits entity names to find similar partitions.

From the matching phase point of view, each matching system uses its own matching strategy which exploits linguistic and structural features of ontologies. Some of these systems make use of existing matching strategies, such as TaxoMap (using the Falcon-AO match strategy) and the Unbalanced OM approach utilizing the similarity flooding algorithm. More specifically, this means that these matching systems do not implement matching strategies specific to this kind of matching, however, they utilize off-the-shelf matching strategies.

It is also worth noting that some matching approaches interlink between the last two phases, i.e. they do not focus on getting local match results for each matching task, but directly construct the final match result. Other matching approaches, like COMA++, first consider each match task as a completely independent match task getting its own local results and then merge or combine these local results to get the final match result.

Future Directions. In this paper we introduced a first conceptual comparison of partitioning-based matching approaches. This will be followed up by an experimental evaluation to determine which combination of approaches works best in which circumstances and to identify necessary areas of improvement.

References

1. A. Algergawy, S. Massmann, and E. Rahm. A clustering-based approach for large-scale ontology matching. In *ADBIS*, pages 415–428. Springer, 2011.
2. H. H. Do and E. Rahm. Matching large schemas: Approaches and evaluation. *Information Systems*, 32(6):857–885, 2007.
3. F. Hamdi, B. Safar, C. Reynaud, and H. Zargayouna. Alignment-based partitioning of large-scale ontologies. In *SCI*, volume 292, pages 251–269. 2010.
4. W. Hu, Y. Qu, and G. Cheng. Matching large ontologies: A divide-and-conquer approach. *DKE*, 67:140–160, 2008.
5. E. Rahm. *Schema Matching and Mapping*, chapter Towards Large-scale Schema and Ontology Matching, pages 3–27. 2011.