Chapter 65
Automatic Schema-Independent Linked Data Instance Matching System

Khai Nguyen
The Graduate University for Advanced Studies, Japan & Ho Chi Minh City University of Science, Vietnam

Ryutaro Ichise
The Graduate University for Advanced Studies, Japan

ABSTRACT

The goal of linked data instance matching is to detect all instances that co-refer to the same objects in two linked data repositories, the source and the target. Since the amount of linked data is rapidly growing, it is important to automate this task. However, the difference between the schemata of source and target repositories remains a challenging barrier. This barrier reduces the portability, accuracy, and scalability of many proposed approaches. The authors present automatic schema-independent interlinking (ASL), which is a schema-independent system that performs instance matching on repositories with different schemata, without prior knowledge about the schemata. The key improvements of ASL compared to previous systems are the detection of useful attribute pairs for comparing instances, an attribute-driven token-based blocking scheme, and an effective modification of existing string similarities. To verify the performance of ASL, the authors conducted experiments on a large dataset containing 246 subsets with different schemata. The results show that ASL obtains high accuracy and significantly improves the quality of discovered coreferences against recently proposed complex systems.

1. INTRODUCTION

Instance matching (aka entity reconciliation, entity resolution, or record linkage) (Winkler, 2006) is the process of detecting coreferent instances, which describe the same object. One prominent application of instance matching is data integration. Since data are created independently in many repositories, gathering information from multiple sources can greatly improve the completeness and diversity of the objects

DOI: 10.4018/978-1-5225-5191-1.ch065

of interest. Detecting coreferent instances is indispensable for achieving perfect integration quality. In linked data, instance matching also plays an important role in the data publication process. The newly published instances should be linked to their existing coreferent instances on the web of linked data. In other words, instance matching, together with other tools, allows linked data instead of just enriched data to become closer to the vision of the semantic web (Jain, Hitzler, Yeh, Verma, & Sheth, 2010). Instance matching in linked data (Ferrara, Nikolov, & Scharffe, 2011) is also considered as a representative of link discovery, because the result of matching can be used to generate the owl:sameAs[1] links, which are conventionally used to declare the coreferences.

The major challenges of instance matching are the ambiguity of instances and the inconsistency between different repositories. The first challenge is the natural heterogeneity of real-world objects (e.g., Tokyo, Tokyo Station, Tokyo Imperial Palace). The second challenge is the different schemata, in which the attributes of objects are declared through arbitrary properties (e.g., 'name' and 'label' co-describe the same attribute). In linked data and other sorts of web-based data, some of the challenges are even harder compared to other forms of structured data because most resources are contributed by the prolific Internet community. On the one hand, the linked data resources provide excellent benefits thanks to the plentifulness of the data. However, on the other hand, they increase the chance of having more instances that refer to very similar objects. Many linked data sources are constructed by many users or from crowdsourced data. Consequently, the inconsistencies of schemata become more complex. For instance matching on linked data, it is more difficult to construct all correct property mappings between given schemata. However, the difficulty can be solved by a schema-independent system. Therefore, schema-independent instance matching systems, which can work on repositories with any schema, have the highest generality.

Many years of investigating a perfect solution for linked data instance matching have resulted in considerable achievements, but not yet the optimal solution. Numerous studies have been published, and they vary from manually operated to semi-automatic and automatic systems. To use manual systems (Volz, Bizer, Gaedke, & Kobilarov, 2009, Ngomo & Auer, 2011, Li, Tang, Li, & Luo, 2009), the user needs to provide matching specifications (e.g., property mappings, similarity measures). Semi-automatic systems try to reduce the user involvement by suggesting a specification (Lyko, Höffner, Speck, Ngomo, & Lehmann, 2013) or by requiring a small number of labeled data (Ngomo, Lehmann, Auer, & Höffner, 2011, Isele & Bizer, 2013). Recently, studies on automatic approaches have increased because of their generality. Existing automatic systems can be categorized into three families: unsupervised learning of specifications (Nikolov, d'Aquin, & Motta, 2012, Ngomo & Lyko, 2013); probabilistic matching (Niepert, Meilicke, & Stuckenschmidt, 2010, Suchanek, Abiteboul, & Senellart, 2011); and similarity-based matching with statistical estimation of property mappings (Araujo, Tran, DeVries, Hidders, & Schwabe, 2012, Nguyen, Ichise, & Le, 2012a). The first two families have a limitation in scalability, because they either repeatedly browse the data or memorize all computations. Meanwhile, the third one is more scalable, due to its simple architecture. One drawback of previous systems in this third family is the low accuracy on large data. However, with its advantage in scalability, this is still one of the most promising solutions.

In this paper, we present ASL (automatic schema-independent interlinking), a system classified into the third family, which is the automatic approach. ASL performs a three-step matching process. The first step is the detection of property mappings by using a value-overlapping measure on the attributes of instances. The next step is a token-based blocking procedure that quickly discards dissimilar candidate pairs of instances. Finally, the last step verifies the remaining pairs by estimating their similarity.