A survey on web data linking

Manel Achichi, Zohra Bellahsene, Konstantin Todorov

LIRMM / University of Montpellier, France
firstname.lastname@lirmm.fr

ABSTRACT. *Data are being published continuously on the web in a decentralized manner leading to a web of heterogeneous data. Given the large amount of published data, access to relevant information becomes difficult, hence the need to interconnect these data. In this paper, we propose a survey on approaches and tools addressing the data linking problem. The particularity of this survey is that we consider the linking processes as a pipeline composed of pre-processing, main matching and post-processing phases and we review the different techniques applied on each of these three steps in service of the global linking task. The actual task of linking two data instances is certainly at the core of this process; however, what happens before and what happens after this task is performed, is of crucial importance for the effectiveness and the efficiency of a data linking tool. One of the important contributions of this paper lies in the organization of the approaches and tools in a (pseudo-) taxonomy, with respect to the three major steps of the matching process (pre-processing, data matching and post-processing), splitting them further into several categories according to the tasks that each approach adresses and finally – according to the techniques that are applied. We additionally consider a fourth, multi-step category of methods – those that act on more than one step of the matching process (they can be found, on multiple leaves of our taxonomy). Finally, we describe and compare different state-of-the-art approaches and tools according to a set of criteria.*

RÉSUMÉ. *Les données sont publiées en continu sur le web et ce de manière décentralisée conduisant à un web de données hétérogènes. Au vu de l'énorme quantité de données publiées et de leur hétérogénéïté, se pose la difficulté d'accéder efficacement à l'information pertinente d'où la nécessité d'interconnecter ces données. Dans cet article, nous proposons un état de l'art des méthodes et outils traitant du problème de liage de données. La particularité de cette étude est que nous considérons le processus de liage comme un pipeline composé de trois phases : 1) pré-traitement, 2) appariement d'instances de données et 3) post-traitement. La tâche proprement dite d'appariement d'instances de données est certainement au cœur de ce processus. Cependant, ce qui se passe avant et ce qui se passe après cette tâche est d'une importance cruciale pour l'efficacité d'un outil de liage de données. Parmi les contributions importantes de cet article il y a la proposition d'une organisation des approches et outils dans une (pseudo-)taxonomie, en fonction des trois grandes étapes du processus. Cette classification comprend plusieurs catégories en fonction des tâches que chaque approche utilise et selon les techniques qui y sont appliquées. Nous considérons par ailleurs une quatrième catégorie de méthodes appelée multi-étapes comprenant les méthodes agissant sur plus d'une étape du processus de liage (ces méthodes*

peuvent être trouvées sur plusieurs feuilles de notre taxonomie). Enfin, nous proposons également une analyse comparative selon plusieurs critères des différentes approches et outils existants dans ce domaine.

KEYWORDS: *web of data, data linking, instance matching.*

MOTS-CLÉS : *web de données, liage de données, appariement d'instances.*

DOI:10.3166/ISI.21.5-6.11-29 © 2016 Lavoisier

1. Introduction

Although sharing data on the web has never been easier, it is yet quite difficult to quickly access relevant information. Data heterogeneity and distribution being at the origine of this problem, structuring and interconnecting data promises to help unlock the potential of the web as a source of information and knowledge. The semantic web emerged with the aim to give meaning to data so that they can be interpreted by both machines and humans, expanding the web of documents that we know towards a new web – the web of data. The RDF (Resource Description Framework) standard was proposed to describe and structure resources (real-world entities) as <subject-predicate-object> triples and ontologies are used in order to annotate these data with commonly agreed on terms, allowing to reason and infer new relations. Since this process is entirely decentralized and strongly human biased, it is more than common that multiple resources, referring to the same real-world entity, exist across RDF datasets published on the web. It is, therefore, crucial to establish links of equivalence between these resources, given by an "owl:sameAs" statement, in order to interconnect datasets, but also connect resources *via* links of different types (other than the equivalence relation) in order to enrich the existant knowledge.

We can see the importance of this process by taking as an example the Doremus project[1], where data linking is a central task. After transforming data to RDF from their original format (mostly MARC[2] and its variants) by following the Doremus model (Choffé, 2016), we end up with several datasets describing musical works and events. On the one hand, equivalent resources (e.g., musical works) across the institutional datasets have to be linked together; on the other hand, links to established knowledge graphs on the web (such as DBPedia) have to be provided. Given the size of the datasets and the number of properties to compare, this task cannot be handled manually and automatic approaches have to be applied.

The *data linking* problem is known to the computer science community for long years. It has been addressed in several research fields such as relational databases and natural language processing (NLP) and referred to by using terms such as "entities resolution", "co-reference resolution" or "duplication identification". All these expressions refer to the problem of discovering, from a given source type (text do-

1. http://www.doremus.org/
2. http://www.loc.gov/marc/

cuments, web pages, databases), different entities that identify one and the same real world object. In the semantic web field, the terms "*instance matching*", "*data linking*", or "*link discovery*" are used to refer the same problem. It aims at determining if two given resources refer to the same real world object or not. This paper provides a survey on data linking tools/approaches at the instance level.

Several approaches to data linking have been proposed and many tools have been developed to deal with this issue through the past years. These approaches/tools have been surveyed in (Ferrara, 2013), which classifies data linking techniques along the criteria of *granularity*, *type of evidence*, and *source of evidence*. The first criterion concerns identifying at which level the matching process is performed. The authors identify three main categories: **(1)** ***Value matching***: consists in identifying equivalence between property values (labels) of instances; **(2)** ***Individual matching***: consists in deciding if two instances represent the same real world object based on their property values; **(3)** ***Dataset matching***: consists in aligning two datasets based on equivalences between their individuals. The second criterion consists in identifying the type of information, on which the linking methods rely. The authors define two main categories: **(1)** ***Data level***, exploiting information about the individuals and their property values, and **(2)** ***Knowledge level***, exploiting information relevant to the ontological schema and/or the external sources (linguistic resources for instance). Regarding the third criterion, the authors define two categories: **(1)** ***Internal***: expressing the fact that the techniques used for linking exploit information contained in the datasets being matched; **(2)** ***External***: expressing the fact that the techniques rely on information contained in external sources. Overall, the authors classify data linking techniques according to these three criteria. Then, they classify data linking tools according to the surveyed techniques.

In the current survey, we take a different stance. We are interested in the linking process from a global perspective. The actual task of linking two data instances is certainly at the core of this process; however, what happens before and what happens after this task is performed is of crucial importance for the effectiveness and the efficiency of a data linking tool. We consider three main steps in the linking workflow: preprocessing, instance matching and post-processing. In this paper, when we talk of data linking approaches, we refer to any approach that is situated on either of these three levels, but also to approaches that perform more than one of these steps. To the best of our knowledge, no study has focused on classifying data linking approaches according to these three levels by considering the linking process as a whole.

Outline. The rest of this paper is organized as follows. In the following section, we introduce the data linking problem by providing a formal definition and the general workflow of data linking. Section 3 overviews a number of state-of-the-art approaches, which operates on one step for dealing with this problem. Section 4 describes the multi-step approaches that operate on more than one of the three levels of interest. We compare and discuss these approaches/tools in Section 5. We conclude in Section 6.

2. The data linking problem

Data are being published continuously on the web in a decentralized manner leading to a web of heterogeneous data, containing duplicates, incomplete information and often even errors. Data linking promises to address this issue and thus considerably improve the quality of web data, facilitating the access to distributed and decentralized information on the web. In this section, we first give our understanding of the data linking task and then present the general workflow of data linking.

2.1. Definition

In the semantic web field, the data linking task is defined as the process of establishing a relation between two resources coming from two distinct datasets, or in other words, declaring a triple that has its subject in one dataset and its object in another. We are interested in a specific kind of data linking that aims to connect *identical* instances through an equivalence relation. We will refer to data linking as the process of comparing two instances of two corresponding classes, across two datasets. The outcome of this process is the establishment of an equivalence link between the resources together with a degree of confidence of this assertion. As mentioned in the introduction, two instances that are found to refer to the same real world object are declared as being equivalent by linking their URIs in an “owl:sameAs" statement of the kind <URI_1, *owl:sameAs*, URI_2>, where URI_1 and URI_2 are the URIs of the two instances.

Formally, the linking process is defined as follows.

- Let ***g*** be an RDF graph which is defined by a set of triples of the kind ***t*= <*s*, *p*, *o*>**, where ***s*** is a subject (an URI or a blank node), ***o*** is an object (an URI or a literal) and ***p*** is the relation (an URI) between ***s*** and ***o***. An RDF graph contains a set of instances, that we denote by I. For instance, the graph given in Figure 1 presents the description of the composer Ludwig van Beethoven in the DBPedia [3] dataset. It is composed of three triples. The first one has as object "foaf:Person", the type of the resource. The second one has as object the literal "*Ludwig van Beethoven*", the name of the resource. The third one has as object an external link <https://en.wikipedia.org/wiki/Ludwig_van_Beethoven>, to a web page which describes the resource.

- For the linking task between two instances i_1 and i_2 across two sets of instances I_1 and I_2 from two different RDF graphs, g_1 and g_2, respectively, we define the function:

$$f : I_1 \times I_2 \rightarrow [0, 1]$$

$$(i_1, i_2) \mapsto s,$$

Where $s \in [0, 1]$ and $s = f(i_1, i_2)$. The function f produces a similarity value s measuring the proximity between two RDF resources i_1 and i_2. These resources are

3. http://wiki.dbpedia.org/

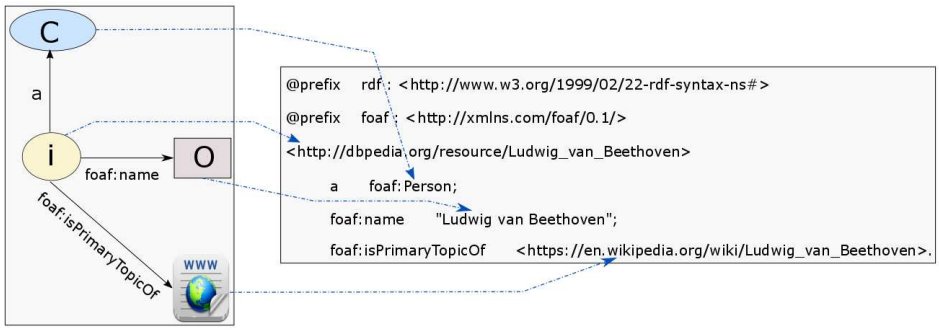

Figure 1. Example of an RDF graph in Turtle format

linked together (i.e., i_1 and i_2 represent the same real world object) if s is greater than a given threshold $\sigma \in [0, 1]$.

2.2. Data linking workflow

The linking process is composed of three main steps: pre-processing, matching, and post-processing (see Figure 2). The pre-processing step aims at preparing data for linking by representing them in a manner that allows for the comparison of data instances coming from different datasets. More than that, this step also aims at reducing the space where to look for potential linking candidates by identifying key properties and equivalent classes of instances across datasets, assuring computational efficiency on large scale. The instance matching task provides an assertion on the degree, to which data instances can be considered as referring to the same real-world object. Finally, the post-processing step further improves the linking results by filtering out erroneous matches or inferring new ones.

Note that we make a distinction between linking *tasks*, linking *techniques* and linking *approaches/tools*. A *task* is defined as a subproblem of data linking, originating at a particular need and identified by a particular result that has to be achieved in service of the linking process. For example, “search space reduction" is a particular task aiming at reducing the number of instances to be compared. A *technique* is understood as the means to perform a task – for example, the "clustering" is used to perform search space reduction. Finally, an *approach*, or a *tool* refers to an engineering artefact that performs a given (set of) task(s) by applying and combining a number of techniques.

3. Review of single-step methods

Here, we take a look at approaches and tools that are devised to handle one single step of the workflow.

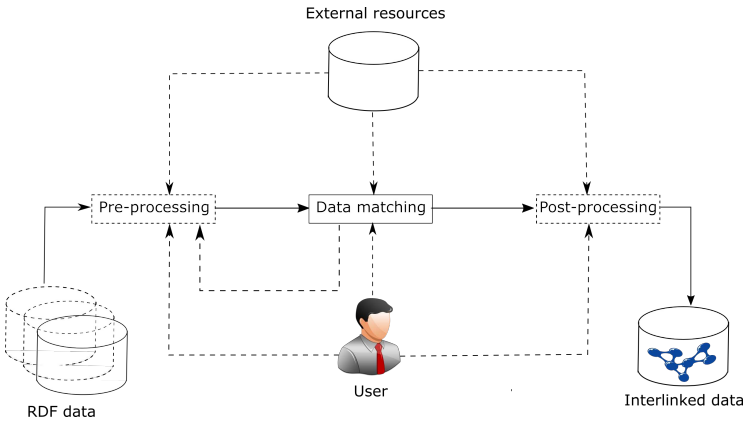

Figure 2. Data linking workflow, adopted from (Ferrara, 2013)

3.1. Pre-processing step

As discussed in the previous section, the pre-processing step consists in preparing the input datasets for linking. It aims to reduce the research space or to represent the instances in an appropriate manner for comparison.

1. **Reducing the research space**. Comparing all pairs of instances through all their properties is costly and time consuming. An instance, being described by a set of properties, a number of comparisons of the different property values has to be performed for each pair of instances. It is, therefore, desirable to reduce both the number of properties and the number of instances prior to the data matching task.

- **Instances number reduction** is usually based on *clustering* techniques, assuming that instances sharing some properties (keywords for example) may be potentially identical. Equivalence links occur in each cluster. In other words, the number of inter-cluster *owl:sameAs* links is fixed at *zero*. Another way to avoid comparing all instances is to specify the conditions that all data items must fulfill in order to be compared. These conditions are specified by a human expert and they concern either the class to which the resources must belong and/or the properties they must have.

- **Properties number reduction** is based on *keys identification*, which consists in discovering sets of properties that uniquely identify the resources. This is a way to avoid comparing all property values to decide whether two resources are equivalent or not. The comparison task is applied on the data matching step where instances sharing the same values for a key, which may consist of one or more properties, are considered as equivalent.

2. **Instance representation**. Various kinds of transformations on the original data can be applied to make the resources comparable. These transformations are linguistic analysis to discover links between entities described in different languages and they are numerical to represent entities in vectors based on computed scores. The linguis-

tic representation often uses external resources, such as BabelNet (Navigli, 2012) or WordNet (Miller, 1995), to perform the data transformation.

In what follows, we present approaches based on keys identification that act on the pre-processing step only.

Atencia *et al* (Atencia, 2012) proposed an approach which is based on two measures for keys detection. The approach consists in determining whether a set of properties is a key for the particular data set. However, because of erroneous and redundant data, the authors extended the key definition to that of a *pseudo-key*, defined as a set of properties that identifies *most* of instances in a RDF dataset. For this purpose, the measures of discriminability and support are introduced. The *discriminability* is defined as the ratio between the number of instances sharing identical values (for this key) to the support of the key. The *support* is the proportion of individuals having all the predicates of the key instanciated. A set of properties is considered as a pseudo-key if its discriminability value is greater than a given threshold.

Similarly to (Atencia, 2012), the notion of *almost-key* is proposed by Symeonidou *et al.* (Symeonidou, 2014) to describe sets of properties that fail to be keys due to few exceptions. To filter data, the authors propose to discover first the maximal non keys and use them to derive the minimal keys. The approach is implemented in the tool SAKey (Scalable Almost Key discovery) and proceeds in three main steps. The first step allows to eliminate irrelevant sets of properties. Based on pruning strategies, the second step allows for the discovery of $(n + 1)$ maximal non-keys, i.e., $n + 1$-non keys that are not subsets of other $n + 1$-non keys for a fixed n. In the third step, the algorithm derives the almost keys from the set of $(n + 1)$ non-keys, i.e., all the sets of properties that are not maximal $(n + 1)$-non keys are n-almost keys.

In the same way, KD2R (Symeonidou, 2011) discovers the maximal non-keys to infer keys. The authors introduce the notion of *undetermined keys* to designate a set of properties that are not a non-key. There are at least two instances that share same values for a subset of undetermined keys. The remaining properties are not instantiated for at least one of the two instances. In other words, an undetermined key defines a set of properties that cannot be considered neither as keys neither as non-keys due to the lack of information. Therefore, the authors introduce the optimist and the pessimist heuristics, implying the consideration of this set of properties as a key or not, respectively. To determine the set of keys in RDF datasets, the algorithm starts by presenting the instances of a given class in a structure called *prefix-tree*. This structure is used to discover the set of maximal undetermined keys and the set of maximal non-keys. The minimal keys are then derived from the previous sets. The algorithm is iterative and is applied for each class of a given ontology.

Unlike SAKey (Symeonidou, 2014), ROCKER (Soru, 2015) considers two resources as distinguishable with respect to a set of properties P even if they share one (or more) object(s) for each $p \in P$. It relies on a scoring function to compare sets of properties and allows the discovery of keys or almost-keys within a given threshold (i.e., a set of properties is a non-key if its score is less than 1). The score function

expresses the number of subject resources that are distinguishable with respect to a set of properties. The authors combine the characteristics of the refinement operator (i.e., prune the refinement tree) with the key monotonicity property to obtain a time-efficient approach for detecting keys. In particular, using monotonicity of keys allows to check for the existence of keys as well as decide on nodes that need not be refined.

3.2. Data matching

The *data matching* step is at the very heart of the linking problem. It aims at finding instances referring to the same real world object by using appropriate instance similarity measures. In the semantic web field, the data matching process operates at different **levels** depending on which piece of information the comparison between instances is done. In fact, we have identified in the literature two levels of comparison -an *intensional* and an *extensional* levels (see Figure 3). In this context, the *intensionality* means information that describes implicitly a resource. On the other hand, the *extensionality* covers information that describes explicitly a resource. For the linking task, a question is raised around this issue: ***between which pieces of information the comparison is done?*** Whether at *extensional* or *intensional* levels, resources can be compared based on several types of information that define them.

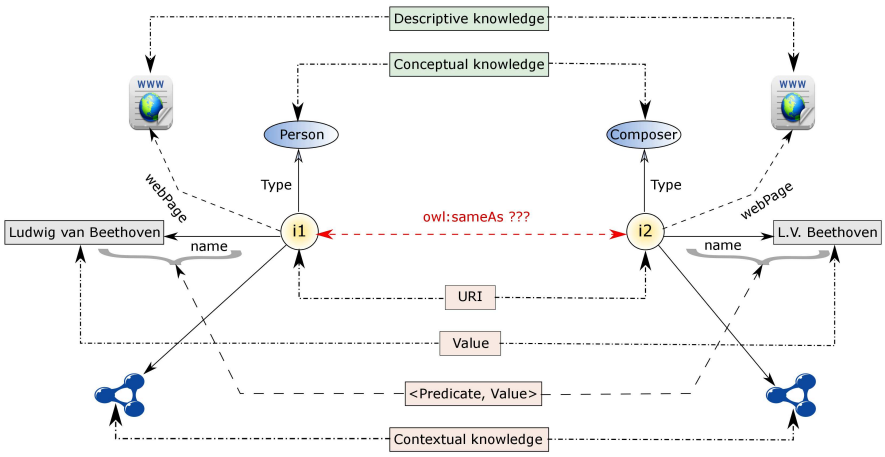

Figure 3. Data matching levels

1. Linking at **extensional level** consists in generating equivalence links based on explicit information. In this case, links can be established, between resources, comparing their:

- ***URIs***: The idea is to interlink entities comparing their URIs. Links can be established if the last fragments of their URIs are identical. For example, we find that the two resources I_1= http://dbpedia.org/page/Ludwig_van_Beethoven and I_2=

http://yago-knowledge.org/resource/Ludwig_van_Beethoven in DBpedia and Yago represent the same entity.

- ***<Predicate, Value> pairs***: To interlink two resources, some approaches compare their literals attached to given properties. In other words, two resources are linked together if they share the same values for all or a subset of their properties identified as a key (see Subsection 3.1).

- ***Values***: Here, the idea is to compare the literals, which are at a distance [4] $n = 1$ to each of the two resources (the literals are directly attached to the resources to be compared) regardless to the properties.

- ***Contextual knowledge***: In some cases, especially in the presence of property chains, information related to *neighboring* resources in the RDF graph is used, because literals directly related to them are not sufficient to decide whether these resources are identical. In other words, two entities are compared based on the literals which are at distances 1 to n to the resources with $n > 1$ in the RDF graph.

2. Linking at **intensional level** generates equivalent links based on information derived from instance's *descriptive knowledge* or *conceptual knowledge*.

- ***Descriptive knowledge***: Generally, approaches relying on this type of information exploit the words contained in the descriptions of two resources to decide whether they are likely to be linked. This means that two entities sharing the same words in their description (for instance, the corresponding wikipedia pages) are considered to be linking candidates.

- ***Conceptual knowledge***: The types of instances carry important knowledge of their similarity and are often used to filter out candidates both in the pre-processing and the matching steps. Indeed, the semantic equivalences between data items can be discovered only if these items belong to the same class (i.e., their concepts match). Many approaches have been proposed where the problem is referred to *concept matching*. Note that this paper is devoted to survey only instance matching tools and approaches. Hence, no approach will be assigned at this category.

Note that some categories will not be illustrated in the remainder of this subsection as is the case of descriptive knowledge-based approaches that may perform other steps of the overall linking process (see Section 4). In the following, we briefly describe approaches proposed in (Raimond, 2008), (Matthew, 2009) and (Jaffri, 2008) acting only at this stage and they can be based on several pieces of information for comparison task.

To illustrate the interlinking process based on ***contextual knowledge*** and on the ***similarity between values***, we present in this paper LD-Mapper (Raimond, 2008), an interlinking system of musical datasets, which is based on the similarity between the resources in the RDF graph and the similarity of their *neighbors*. The system starts

4. A distance is defined over the nodes (resources) of a (RDF) graph as the minimal number of edges connecting two nodes (resources).

by computing the *similarity values between literals* of all pairs of resources. Then, it combines a measure (aggregation of all the similarity values between resources pairs) for each graph mapping. The graph mappings are all the possible combinations between the resources of the two graphs. Finally, the mapping of which the similarity value is the largest will be selected by the algorithm.

In the same category, the approach proposed in (Matthew, 2009) is based on ***contextual knowledge*** and on the ***similarity between values***. It is a user profiles interlinking approach of different social networks such as Facebook, Twitter and MySpace. The idea consists in constructing RDF graphs from XML files of the social networks. These RDF graphs will then be interconnected based on the *user identifiers* of each social network. Data linking uses several similarity measures based on the graphs in order to create *owl:sameAs* links.

In the conference/university field, a coreference resolution system called RKB-CRS (Jaffri, 2008) has been introduced, based on the ***similarity between values***. It consists in establishing a list of equivalent URIs (resources). For each datasets pair, a new program has to be written. This program selects the resources to be aligned and it compares them using *similarity measures*. RKB-Explorer [5] uses this system to store, manipulate and reuse coreference data.

3.3. Post-processing methods

In this sub-section, we present approaches operating only once data is interlinked. These methods aim to evaluate the existing links in a data set.

A **partitioning method**, proposed by (Guizol, 2013), detects the existing erroneous links in a bibliographic knowledge base where each bibliographic record (i.e., a document describing a book for instance) is connected to one or several authority records (i.e., where each of them describes a person who edited or wrote the book). The main idea is to discover if these existing links are correct. The concept of partition, for a set of objects, is defined as a set of classes such that each object belongs to only one class. Their global method proceeds as follows:

1. Construct contextual authorities where each is composed of an authority record with one of the bibliographic records pointing to it. It corresponds to an author with one written book.

2. Partition the set of contextual authorities according to different criteria (title, date of publication, domain). Each partition makes sense from the point of view of the respective criteria.

3. Aggregate criteria to decide whether these contextual authorities represent or not a same person using two proposed semantics.

5. http ://www.rkbexplorer.com/explorer/

4. Multi-step methods

Multi-step methods, as mentioned above, combine two or three steps of the data linking workflow.

The SERIMI system (Araujo, 2011) operates in two steps: selection and disambiguation. In the selection phase, SERIMI extracts, for each instance in dataset A its label (identifier) and looks for instances in the dataset B that have a similar label according to a given threshold (i.e., retrieve target candidate resources with a string similarity greater than 70%, as fixed by the authors). This step outputs a set of *pseudo-homonym* resources for each source resource. As distinct instances may share the same label, the disambiguation phase will allow to select the appropriate instances among the set of *pseudo-homonym* resources. Indeed, the disambiguation phase consists in filtering the instances found in the dataset B and keeping only those that identify the same real world object as the resources in the dataset A. As the class of the target resources is not known, SERIMI selects the resources that belong to the *class of interest* to filter instances in the *pseudo-homonym* sets. The authors define the concept of *class of interest* as a set of attributes that instances may share in common. The idea is to classify a target resource as belonging to a given *class of interest* by comparing it to all the other *pseudo-homonym* sets. Finally, the system selects all resources with a score greater than a given threshold.

To tackle the multilingualism problem, Lesnikova *et al*. (Lesnikova, 2014) proposed a linking method using indexing and NLP techniques. The method consists in creating a document (accumulation of data collected in the graph traversal) for each URI. All documents are automatically translated into a pivot language before computing similarity between them. The same authors proposed a similar approach, which relies on the use of a ***multilingual*** lexical resource (BabelNet (Navigli, 2012)) instead of a machine translation technique (Lesnikova, 2015). Once the documents are constructed, the algorithm replaces each term by the corresponding identifier (ID) from BabelNet. There may exist many senses (IDs) per term. If it is the case, word sense disambiguation techniques are applied in order to select the most appropriate sense. After preprocessing, the authors index the documents by using the vector space model to represent the documents as word feature vectors. They compute a set of similarity values between pairs of documents by using the standard term weighting scheme TF-IDF and applying the cosine similarity. Resources are then linked based on the produced similarity scores between the documents.

Indexing techniques can also be used to reduce the number of instances comparison. Indeed, Rong *et al*. (Rong, 2012) developed a system, which extracts in the first step literal information from entities. The literal information $l = \{l_1, l_2, ..., l_n\}$ is similar to a document generated from an instance. Then, it uses a vector space model to represent this information. An inverted ***index*** is built for instances of some keywords in their descriptions. The instances sharing the same key words in their index are considered to be linking candidates. Then, the algorithm computes the similarity vectors for the candidate instance pairs. The authors propose a ***feature vector*** of si-

milarity metrics. To train a ***binary classifier*** based on the similarity vectors, they are labeled into two classes: *non-matching* and *matching*. The existing links in the LOD cloud can also be used to train the classifier. The authors evaluate the proposed approach on the datasets of OAEI 2010 showing that the approach performed better than the other paricipants.

To prepare the instances for the linking task, RiMOM-IM (Shao, 2016) unifies languages and/or formats in which data are expressed, removes stop words, or computes the TF-IDF of words composing the predicate values. After that, RiMOM-IM applies a blocking technique which consists in using inverted indexing to generate candidate sets and unique instance sets. It takes the predicate and the top five words of the object (ordered by TF-IDF values) as index keys of instances. For each pair in the candidate set, it computes similarities over all aligned predicates and then aggregates them to get the final matching score of two instances. It iteratively selects the pair with the highest score (above a given threshold) as the aligned pair. It will be used to infer new candidate pairs without computing the similarities until no new aligned instances are generated, by using two strategies: *unique subject matching* and *one-left object matching* (see (Shao, 2016) for detailed descriptions).

To avoid exhaustive pairwise comparisons of instances, a blocking step is also applied in (Kejriwal, 2015) by grouping together properties. Unlike RiMOM-IM, two instances are in the same cluster if they share tokens of the labels of any two properties that were grouped together. The authors propose to apply the block purging algorithm which eliminates clusters larger than a threshold value, with the premise that such clusters are the result of stop-word tokens. Once the candidate set is generated, the system tries to ensure that only compatible instance pairs are evaluated by generating restriction sets (matching classes and properties between two RDF files). Each compatible instance pair is represented by the help of the boolean model where the values are equal either to 1 if the instances share a common token or to 0 otherwise. A binary classifier is trained on these data and applied to discover new links.

To prune the research space, Silk (Volz, 2009) implements indexing and entity preselection methods. The pre-selection consists in finding a limited set of target entities that are likely to match a given source entity. All target resources are indexed by one or more specified property values (most commonly, their labels). The `rdfs:label` of a source resource is used as a search term into the generated indexes and only the first target resources found in each index are considered as link candidates for matching. This strategy does not ensure the identification of all equivalent resources in the target dataset. Silk is based on user link specifications (Silk-LSL). In other words, it features a declarative language for specifying which types of RDF links should be discovered between data sources and which conditions entities must fulfill in order to be interlinked.

As SILK, LIMES (Ngomo, 2011) is configured by using a link specification language. It is based on the mathematical characteristics of metric spaces to compute the similarity between instances. In particular, it utilizes the *triangle inequality* to reduce the number of comparisons and therefore to reduce the time complexity of the map-

ping task, which is one of its major advantages. By these means, LIMES partitions the (instance) metric space by representing each of these portions by an *exemplar* that allows to compute an accurate approximation of the distance between instances based on already known distances. Due to the considerable gain in efficiency provided by the tool, LIMES is capable of linking very large datasets, where other tools usually fail.

Nguyen *et al*. (Nguyen, 2012) propose an approach that also combines preprocessing with data matching. The authors introduce the notions of coverage and discriminability to define a property as a key. The *coverage* of a property is defined as the ratio of the number of instances having that property to the total number of instances. The *discriminability* is the ratio of the number of distinct values for the property to the total number of instances having that property. The instances that have similar literal values for the candidate selection key are linked together.

RDF-AI (Scharffe, 2009) is a semi-automatic tool which acts on the three main data linking steps. It takes as an input two RDF datasets and two XML configuration files and generates as an output either a new dataset resulting from the fusion of the two input datasets, or a list of correspondences between equivalent resources of the two datasets. The XML configuration file specifies the pre-processing operations to perform (consistency checking, properties translation, linking techniques, properties transformation) for each resource. Indeed, the pre-processing step generates two datasets processed by the user operations. The second configuration file describes the post-processing parameters such as the correlation threshold and the fusion parameters. The system includes two similarity computation algorithms: a string matching algorithm and a word relations algorithm. There are two implementations to the latter: a synonyms comparison algorithm based on WordNet and a taxonomical similarity algorithm based on the SKOS vocabulary. Finally, the system checks the inconsistencies (for example, breaking ontology axioms) that may appear as a result of the linkset or of the fused graph.

Like RDF-AI, KnoFuss (Nikolov, 2008) takes as an input two datasets with their respective ontologies by specifying the resources to be compared and the comparison techniques to use. It is also able to merge two datasets using existing ontology alignments in case when the two datasets are described by different ontologies. In this context, *ontology matching* allows to translate the SPARQL queries that are used to select the appropriate instances and the comparable properties from the terms of one scheme to another. In this way, the instances are compared in the same manner as if they were described by the same ontology. Just like (Scharffe, 2009), KnoFuss includes a post-processing step to check the inconsistencies of the merged datasets.

5. Discussion and comparison of the tools and approaches

In this Section, for sake of clarity, we present a classification of approaches and tools depicted in Figure **??**. We organize the approaches and tools in a (pseudo-) taxonomy with respect to the three major steps of the matching process (*pre-processing*,

data matching and *post-processing*), splitting them further into several categories according to the tasks that each approach adresses and finally according to the techniques that are applied. We additionally consider a fourth, *multi-step* category of methods – those that act on more than one step of the matching process (they can be found on multiple leaf-nodes of our taxonomy).

In order to evaluate and compare the data linking tools and approaches, we define several criteria allowing to highlight the specificity of each of them. The criteria are described hereafter:

Domain. Certain tools are developed for datasets of a specific field (music, library or conference) while others are generic. The interest here is whether the domain has an impact on the interlinking results or not.

Input. This criterion specifies the type of input data. Indeed, as we have seen, we are interested in this paper to different data formats. The goal is to address similar issues as interlinking RDF data.

Output. The output data type differs depending on the input data type (RDF data or articles) or on the stage (pre-processing, data matching or post-processing) on which focuses the tool.

Pre-processing. This criterion specifies whether the tool applies a treatment before the interlinking phase.

Data matching. As shown in this paper, some tools have been developed just in order to identify keys among a given dataset. Indeed, even if our work focuses on interlinking tools, those presented in this paper were not necessarily conceived to discover equivalence between entities. This criterion specifies whether the tool performs the linking task or not.

Post-processing. This criterion specifies whether the tool applies a treatment after the interlinking phase.

Techniques. This criterion specifies the techniques used by the tool to be compared.

Multilingualism. Any entity matching tool must necessarily dealing the multilingual RDF datasets. In fact, nowadays, RDF data are expressed in 474 languages [6] on the LOD. Therefore, the lingual heterogeneity occurs mainly at the value level. This criterion specifies whether the tool distinguishes the multilingual aspect between the RDF data.

Automation. In addition to the results that the linking tools produce, they should also be compared according to their degree of automation. Indeed, the user intervention may be required before (configuration stage) or after (validation stage) the entities are linked. This allows to give increased costs with respect to the computing time. This criterion specifies the degree of automation of the compared tools.

6. http://stats.lod2.eu/languages

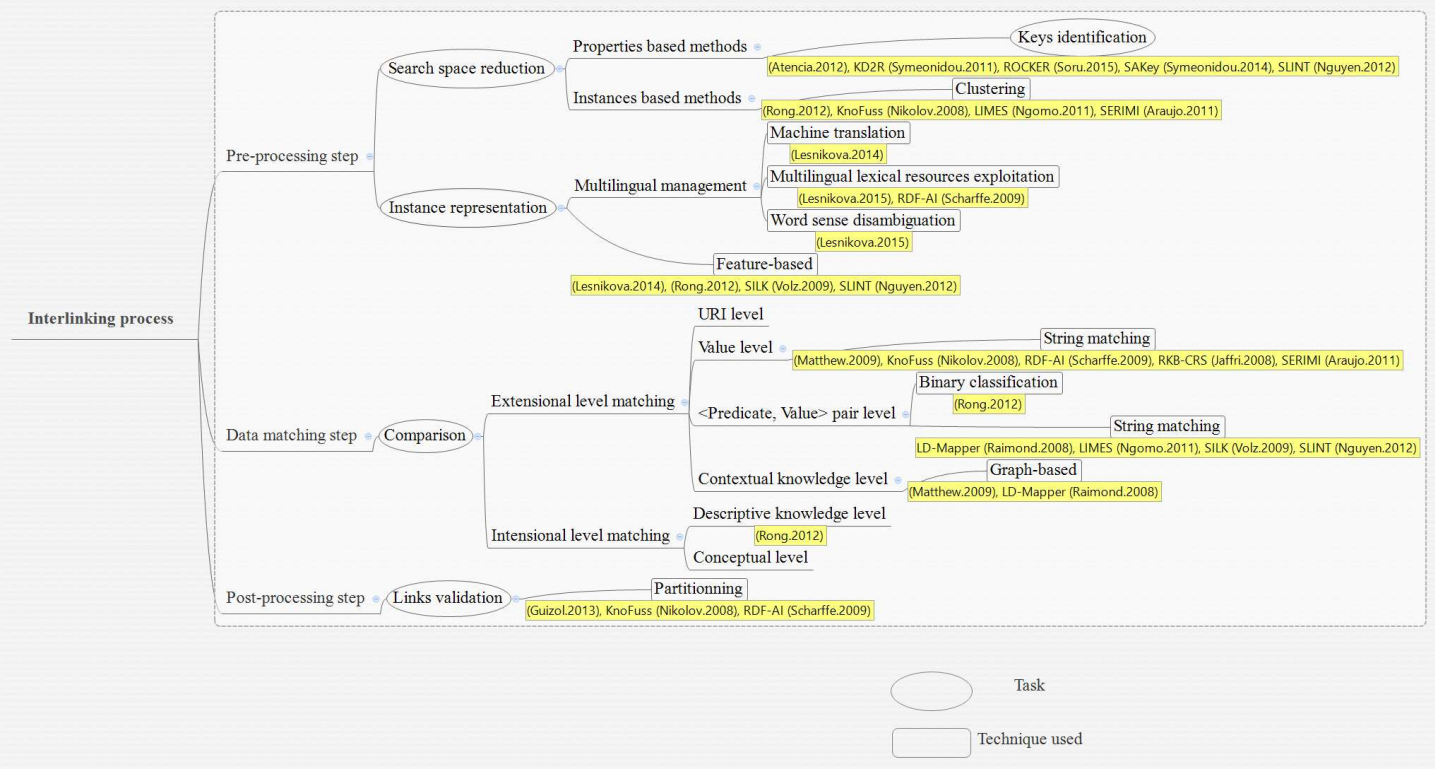

Figure 4. Classification of tools/approaches according to the three main data linking steps, to the tasks and the techniques

Table 1. Summary table of the main linking tools

Ref	Tools	Domain	Input	Output	Pre-processing	data matching	Post-processing	Tasks/techniques used	Multilingualism	Automation
(Symeonidou, 2014)	SAKey	LOD	RDF	Keys	✓	✗	✗	Keys identification	✗	Automatic
(Symeonidou, 2011)	KD2R	LOD	RDF	Keys	✓	✗	✗	Keys identification	✗	Automatic
(Soru, 2015)	ROCKER	LOD	RDF	Keys	✓	✗	✗	Keys identification	✗	Automatic
(Araujo, 2011)	SERIMI	LOD	RDF	Linkset	✓	✓	✗	Clustering String matching	✗	Automatic
(Volz, 2009)	SILK	LOD	RDF Config file	Linkset	✓	✓	✗	Feature-based String matching	✗	Semi-automatic
(Ngomo, 2011)	LIMES	LOD	RDF Config file	Linkset	✓	✓	✗	Clustering String matching	✗	Semi-automatic
(Nguyen, 2012)	SLINT	LOD	RDF	Keys Linkset	✓	✓	✗	Keys identification Blocking Feature-based String matching	✗	Automatic
(Raimond, 2008)	LD-Mapper	Music	RDF	Linkset	✗	✓	✗	String matching Graph-based	✗	Automatic
(Jaffri, 2008)	RKB-CRS	Conferences-universities	RDF Config file	Linkset	✗	✓	✗	String matching	✗	Semi-automatic
(Nikolov, 2008)	KnoFuss	LOD	RDF	Linkset	✗	✓	✓	Clustering String matching One-to-one filter	✗	Automatic
(Scharffe, 2009)	RDF-AI	LOD	RDF Config file	Linkset Merged data-sets	✓	✓	✓	String matching Machine translation Lexicon exploitation inconsistency checking	✓	Semi-automatic

Table 2. Summary table of the main linking approaches

Ref	Domain	Input	Output	Pre-processing	data matching	Post-processing	Tasks/techniques used	Multilingualism
(Atencia, 2012)	LOD	RDF	Keys	✓	✗	✗	Keys identification	✗
(Lesnikova, 2014)	LOD	RDF	Linkset	✓	✓	✗	Machine translation	✓
(Lesnikova, 2015)	LOD	RDF	Linkset	✓	✓	✗	Feature based Word sense disambiguation	✓
(Rong, 2012)	LOD	RDF	Linkset	✓	✓	✗	Lexicon exploitation Clustering Feature-based	✗
(Matthew, 2009)	Social networks	XML	Linkset	✗	✓	✗	Binary classification String matching	✗
(Guizol, 2013)	Bibliographic knowledge	Bibliographic records	Link validation	✗	✗	✓	Graph-based Partitionning	✗

Tables 1 and 2 provide a comparison between the various RDF data linking tools and approaches (respectively) according to the forementioned critira. Each method is described by its principle, the specificity of its application to a particular field (music, multimedia) and its management of multilingualism.

Summary. Analysis of state of the art of data linking has shown that there are as many *single-step* approaches (approaches that focus on only one step in the overall linking process) as *multi-step* approaches (combination of several steps). However, there are very few works that focus on the *post-processing* step. Furthermore, there is no generic approach to tackle the data linking issue from *pre-processing* to *post-processing* step. Also, the *multilingualism* proves to be an important issue to be taken into account in order to identify the same resources across multilingual RDF datasets and interlink them. However, this study has shown that only few approaches tackle this problem. We note that many of the linking tools require a laborious step of configuration, by specifying manually properties, types, similarity measures and other parameters before launching the tool. Automation of the configuration step appears to be a pressing problem in the field. Finally, while there exist multiple generic linking tools/approaches, there is just a few works dedicated to interlinking specific type of data. In fact, the authors of (Raimond, 2008) seem to be the only ones to propose an approach for linking musical datasets. *Does the domain specificity of data affect the results produced by a generic linking approach*? We leave this as an open question which has to be answered by evaluating linking tools on different domain specific datasets.

6. Conclusion and future work

The purpose of this paper is to provide a survey of existing data linking approaches and tools. We classified these tools and approaches according to the step on which they focus and the techniques used in each step. Finally, we discussed this survey based on several criteria especially their ability to manage multilingualism or their full coverage of the interlinking process from pre-processing step to the assessment of generated links.

In the literature, no existing survey on data linking approaches focused their classification on the multiple heterogeneity aspects that data can have. Indeed, these heterogeneities can be the origin of the linking problem. Hence, an instance matching tool has to be able to deal with them. For example, the challenge of linking *multilingual* data remains largely unexplored. Therefore, our future research is heading in this direction. This leads us to another open problem, consisting in the development of a data linking tool overcoming the multiple heterogeneity aspects that will be identified. Finally, since the current work does not take into account the evaluation of the performance of existing matching systems, more work has to be invested in terms of their engineering aspects, i.e., configuration, runtime and availibility.

Acknowledgements

This work has been partially supported by the French National Research Agency (ANR) within the DOREMUS Project, under grant number ANR-14-CE24-0020.

Bibliographie

Atencia, M., David, J., and Scharffe, F. (2012, October). *Keys and pseudo-keys detection for web datasets cleansing and interlinking*. In International Conference on Knowledge Engineering and Knowledge Management (pp. 144-153). Springer Berlin Heidelberg.

Araujo, S., Hidders, J., de Vries, A. P., and Schwabe, D. (2011, October). *Serimi-resource description similarity, rdf instance matching and interlinking*. In Proceedings of the 6th International Conference on Ontology Matching-Volume 814 (pp. 246-247). CEUR-WS. org.

Choffé, Pierre and Leresche, Françoise. (2016). *DOREMUS: Connecting Sources, Enriching Catalogues and User Experience*.

Ferrara, A., Nikolov, A., and Scharffe, F. (2013). *Data linking for the semantic web.* Semantic Web: Ontology and Knowledge Base Enabled Tools, Services, and Applications, 169.

Guizol, L., Croitoru, M., and Leclere, M. (2013). *Aggregation Semantics for Link Validity.*. In Research and Development in Intelligent Systems XXX (pp. 359-372). Springer International Publishing.

Jaffri, A., Glaser, H., and Millard, I. (2008). *Managing URI synonymity to enable consistent reference on the Semantic Web.*. Proceedings of the 1st IRSW2008 International Workshop on Identity and Reference on the Semantic Web, Tenerife, Spain, June 2, 2008.

Kejriwal, M., and Miranker, D. P. (2015, May). *Semi-supervised Instance Matching Using Boosted Classifiers.*. In European Semantic Web Conference (pp. 388-402). Springer International Publishing.

Lesnikova, T., David, J., and Euzenat, J. (2014, May). *Interlinking English and Chinese RDF Data Sets Using Machine Translation.*. In 3rd ESWC workshop on Knowledge discovery and data mining meets linked open data (Know@ LOD). No commercial editor.

Lesnikova, T., David, J., and Euzenat, J. (2015, September). *Interlinking English and Chinese RDF Data Using BabelNet.*. In Proceedings of the 2015 ACM Symposium on Document Engineering (pp. 39-42). ACM.

Matthew Rowe. *Interlinking Distributed Social Graphs.*. In Proceedings of the WWW2009 Workshop on Linked Data on the Web, LDOW 2009, Madrid, Spain, April 20, 2009., 2009.

George A. Miller. *Wordnet: a lexical database for English.*. Communications of the ACM, 38(11):39?41, 1995.

Navigli R., and Ponzetto S. P. (2012). *BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network.*. Artificial Intelligence, 193, 217-250.

Nikolov A., Uren V., Motta E., and De Roeck A. (2008, September). *Integration of Semantically Annotated Data by the KnoFuss Architecture.*. In International Conference on Knowledge Engineering and Knowledge Management (pp. 265-274). Springer Berlin Heidelberg.

Ngomo A. C. N., and Auer S. (2011). *Limes-a time-efficient approach for large-scale link discovery on the web of data.*. integration, 15, 3.

Nguyen, K., Ichise, R., and Le, B. (2012, November). *SLINT: a schema-independent linked data interlinking system*. In Proceedings of the 7th International Conference on Ontology Matching-Volume 946 (pp. 1-12). CEUR-WS. org.

Raimond, Y., Sutton, C., and Sandler, M. B. (2008). *Automatic Interlinking of Music Datasets on the Semantic Web.*. Automatic Interlinking of Music Datasets on the Semantic Web. LDOW, 369.

Rong, S., Niu, X., Xiang, E. W., Wang, H., Yang, Q., and Yu, Y. (2012, November). *A machine learning approach for instance matching based on similarity metrics.*. In International Semantic Web Conference (pp. 460-475). Springer Berlin Heidelberg.

Scharffe, F., Liu, Y., and Zhou, C. (2009). *Rdf-ai: an architecture for rdf datasets matching, fusion and interlink.*. In Proc. IJCAI 2009 workshop on Identity, reference, and knowledge representation (IR-KR), Pasadena (CA US).

Shao, C., Hu, L. M., Li, J. Z., Wang, Z. C., Chung, T., and Xia, J. B. (2016). *RiMOM-IM: A Novel Iterative Framework for Instance Matching.*. Journal of Computer Science and Technology, 31(1), 185-197.

Soru, T., Marx, E., and Ngonga Ngomo, A. C. (2015, May). *ROCKER: A refinement operator for key discovery.*. In Proceedings of the 24th International Conference on World Wide Web (pp. 1025-1033). ACM.

Symeonidou, D., Pernelle, N., and Saïs, F. (2011, October). *Kd2r: A key discovery method for semantic reference reconciliation.*. In OTM Confederated International Conferences" On the Move to Meaningful Internet Systems" (pp. 392-401). Springer Berlin Heidelberg.

Symeonidou, D., Armant, V., Pernelle, N., and Saïs, F. (2014, October). *Sakey: Scalable almost key discovery in rdf data.*. In International Semantic Web Conference (pp. 33-49). Springer International Publishing.

Volz, J., Bizer, C., Gaedke, M., and Kobilarov, G. (2009). *Silk-A Link Discovery Framework for the Web of Data.*. LDOW, 538.