HAL
open science

Methods for automated dataset interlinking

François Scharffe, Zhengjie Fan, Alfio Ferrara, Houda Khrouf, Andriy Nikolov

To cite this version:

François Scharffe, Zhengjie Fan, Alfio Ferrara, Houda Khrouf, Andriy Nikolov. Methods for automated dataset interlinking. [Contract] 2011, pp.34. hal-00793435

HAL Id: hal-00793435

https://inria.hal.science/hal-00793435v1

Submitted on 22 Feb 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Datalift

Un ascenseur pour les données

ANR Contint – ANR-10-CORD-009

D4.1 Methods for automated dataset interlinking

Coordinator: François Scharffe
With contributions from: Zhengjie Fan (INRIA), Alfio Ferrara (Uni. Milan), Houda Khrouf (Eurecom), Andriy Nikolov (Open University), François Scharffe (LIRMM)

Quality reviewer:	Laurent Bihanic
Reference:	Datalift/2011/D4.1/v0.2
Project:	Datalift ANR Contint ANR-10-CORD-009
Date:	December 24, 2011
Version:	0.2
State:	preliminary
Destination:	public

Executive Summary

Interlinking data is a crucial step in the Datalift platform framework. It ensures that the published datasets are connected with others on the Web. Many techniques are developed on this topic in order to automate the task of finding similar entities in two datasets. In this deliverable, we first clarify terminology in the field of linking data. Then we classify and overview many techniques used to automate data linking on the web. We finally review 11 state-of-the-art tools and classify them according to which technique they use. This work will serve as the basis to design an efficient set of interlinking techniques that will be implemented for the Datalift platform. Deliverable 4.2 will present the specific techniques we develop for the platform.

Document Information

ANR Project Number	ANR Contint – ANR-10-CORD-009	**Acronym**	Datalift
Full Title	Un ascenseur pour les donnes		
Project URL	`http://www.datalift.org/`		
Document URL			

Deliverable	**Number**	4.1	**Title**	Methods for automated dataset interlinking
Work Package	**Number**	4	**Title**	Data interlinking

Date of Delivery	**Contractual**	M12	**Actual**	28-09-2011
Status	preliminary		final ☒	
Nature	prototype ☐ report ☐ dissemination ☐			
Dissemination level	public ☒ consortium ☐			

Authors (Partner)	Author 1 (Partner 1), Author 2 (Partner 2)			
Resp. Author	**Name**	Responsible Author	**E-mail**	author@somewhere.com
	Partner	Partner		

Abstract (for dissemination)	This report is a state of the art on available methods and tools for data interlinking.
Keywords	ontology matching, ontology alignment, evaluation, benchmarks, efficiency measure

Version Log			
Issue Date	**Rev No.**	**Author**	**Change**
October 17, 2011	1	François Scharffe	v0.1
December 24, 2011	2	François Scharffe	v0.2

Table of Contents

1 Problem formulation 5
1.1 The data linking task . . . 5
1.2 Terminological hints and related problems . . . 7
1.3 Mappings and their meaning . . . 8

2 Instance matching techniques 11
2.1 Value matching . . . 12
2.2 Individual matching . . . 15
2.3 Dataset matching . . . 18
2.4 Time-Efficient similarity algorithms . . . 22
2.5 Summary . . . 23

3 Data linking systems 24
3.1 LN2R . . . 24
3.2 ObjectCoref . . . 24
3.3 Okkam . . . 25
3.4 RKB-CRS . . . 25
3.5 LD-mapper . . . 25
3.6 Silk . . . 25
3.7 LIMES . . . 26
3.8 KnoFuss . . . 26
3.9 RDF-AI . . . 26
3.10 Zhishi.links . . . 26
3.11 Serimi . . . 27
3.12 Summary . . . 27

4 Open problems and directions 31

5 Conclusion 33

References 33

1. Problem formulation

In the Semantic Web and in the Web in general, which are increasingly seen as systems where active users produce and consume information consisting not only of documents but also of structured data, a fundamental problem is the comparison and matching of these data and the capability of resolving the multiplicity of data references to the same real objects, by defining correspondences among data in form of data links.

In general terms, *data linking* is the task of determining whether two object descriptions can be linked one to the other to represent the fact that they refer to the same real object in a given domain or the fact that some kind of correspondence holds between them. Quite often, this task is performed on the basis of the evaluation of the degree of similarity among different data describing real objects across heterogeneous data sources, under the assumption that the higher the similarities between two data descriptions, the higher the probability that the two descriptions actually refer to the same object. From an operative point of view, data linking includes also the task of defining methods, techniques and (semi-)automated tools for performing the similarity evaluation task. We call this specific subtask *instance matching*[1].

1.1 The data linking task

Data linking can be formalized as an operation which takes two collections of data as input and produces a collection of mappings between entities of the two collections as output. Mappings denote binary relations between entities corresponding semantically one to each other. The data linking task is articulated in steps as shown in Figure 1.1.

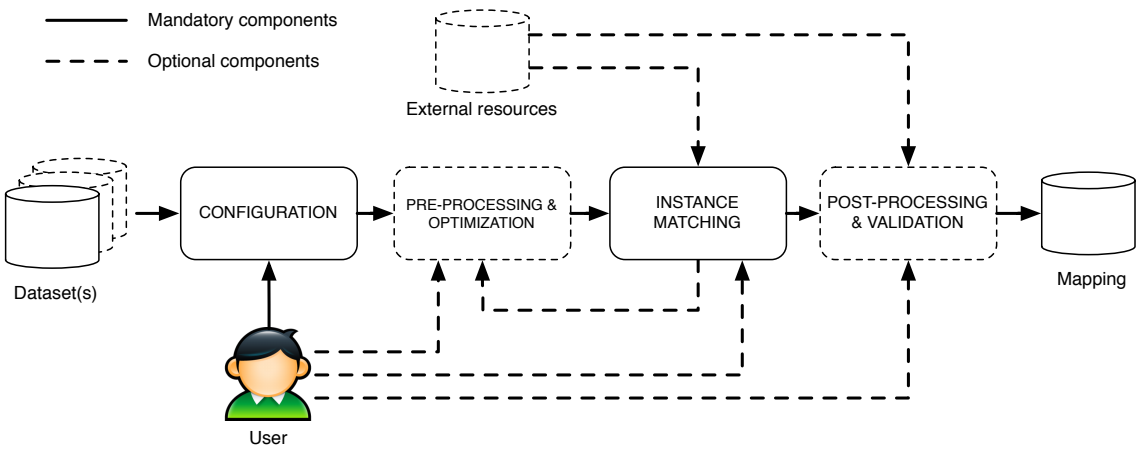

Figure 1.1: The data linking task

The input of the process is one or several datasets. Each dataset is a collection of data representing object descriptions to be linked. The output of the process is a *mapping set*, that is a collection of binary relations (usually referred as *mappings* or *links*) between the object descriptions of the dataset(s) in input. Sometimes, the term *mapping* is used in order to denote the collection of binary relations between object descriptions and the term *mapping rule* is used in order to denote the single correspondence between two object descriptions.

[1] Sometimes however, the expression "data linking" is used instead of "instance matching" to denote only the matching task.

This terminology is more common in the field of ontology matching, where mappings often represent more than a simple correspondence between entities, but a transformation rule holding between two entities [16][2]. The data linking task also involves a user, who has the responsibility of configuring the process and, optionally, interact with the pre-processing, matching and post-processing steps in case of a semi-automatic procedure. Another optional component is given by external resources such as lexical databases and thesauri, reasoning systems or pre-defined mappings that can be used both in the matching and in the post-processing steps as a support for the main process. A typical external resource used for the comparison is a schema level mapping, that is used to determine which data must be compared.

The main steps of the process are more extensively described in the following:

Configuration. The configuration step has the goal of setting up those parameters that are used during the instance matching step in order to execute the comparison between object descriptions. In particular, it is very common to evaluate the object description similarity as a value in the range [0,1] and to set a threshold that denotes the minimum value of similarity needed in order to consider a pair of object descriptions as similar one to the other. Another parameter in some systems is the choice of similarity metrics combination that has to be used in the matching step as well as the list of external resources that may be needed for similarity evaluation.

Pre-processing & optimization (optional) Pre-processing of data is an optional step that can be executed for two main purposes. The first is to transform the original representation of data according to a reference format used for the comparison. A second goal is to minimize the number of comparisons that have to be executed to produce the final mapping set. To this end, the system may implement several kinds of optimization techniques to select the descriptions that have the highest probability to match any given object description.

Matching. In the instance matching step the object description comparison is finally executed according to the metrics chosen in the configuration step. In many cases, more than one type of matching techniques are combined, including for example string/value matching, learning-based matching, similarity propagation. If required, external resources are used in this step to optimize the matching with respect to a pre-defined mapping or to determine the similarity between property values according to existing lexical resources or ontologies (e.g., WordNet, SKOS, OpenCyC). In case of a semi-automatic process, the user interaction is needed in the matching step in order to select the correct mappings or to validate the system result.

Post-processing & validation (optional) The post-processing step is optional and has the goal of refining the matching result according to specific domain or application requirements. A typical post-processing goal is to validate the mappings with respect to a formal definition of consistency for the mapping set, that may include, for example, the fact that two distinct object descriptions cannot be mapped onto the same entity [37]. Another kind of validation is executed with respect to the required mapping set cardinality. In some cases, a mapping set cannot include more than one matching counterpart for each object description.

[2] Another terminological choice is to use the term *alignment* in order to denote the mapping set and the term *correspondence* to denote the mapping.

This requirement can be taken into account either in the instance matching step or in the post-processing step, by defining appropriate heuristics for the deletion of multiple mapping rules and the selection of the best one [9].

1.2 Terminological hints and related problems

The problem of data linking is directly related to many similar problems, and this causes sometimes confusion about the appropriate terminology. Problems similar to data linking can be grouped in two main categories: the first category describes the problems related to the proper activity of *data linking* [1], that is the activity of connecting together different data provided by heterogeneous data sources. In this field, we often talk about *entity matching*, *coreference resolution* or *entity resolution* to denote the activities of evaluating the degree of similarity between pairs of entities with the goal of finding a common reference to a single real object [13, 31, 32]. On the other side, a second category of problems is related to the comparison of data records especially for data cleaning and duplicate detection purposes. In this field, we often talk about *duplicate identification*, *record linkage* or *merge/purge problem*. For both categories of problems many solutions have been proposed. Recent surveys and contributions on this field include for example [33, 6, 3]. The main differences between these two categories of problems and their related solutions can be described according to three main dimensions of analysis: i) the *goal* of the problem; ii) the *object* of the solution, and iii) the *meaning* of the results, as graphically shown in Figure 1.2.

Goal of the problem. The goal of the problem is the final purpose of the comparison task. Some of the problems mentioned above have the goal for example of detecting duplicate records in order to clean a collection of data by capturing errors. This is the case for example of duplicate detection and record linkage, when these techniques are used in the context of data cleaning. Another goal is data integration. In this case, the final purpose is to provide a unique description for each real object mentioned in different data sources. Between these two options, a third goal is the linkage of data. In this case, we admit the presence of different and heterogeneous descriptions of real objects, but we want to create links and relations among these descriptions in order to make explicit their correspondence.

Object of the solution. By object of the solution we mean the kind of data that are taken into account for the comparison. A general but useful distinction concerns techniques and solutions mainly focused on unstructured data, such as plain text or web pages, on one side, and techniques and solutions mainly focused on ontology instances on the other side. Between these two sides of the comparison, it is very important to recall the several methods and techniques that have been conceived in order to work on structured data (usually relational database instances). The main difference between these kinds of solutions is that the assumptions that can be made on unstructured and structured data are different from those that are valid in case of ontological data. For example, in the case of relational data we usually do not deal with multi-valued attributes and the structure of data is explicit. On the contrary, the structure of ontological data is implicit and must to be inferred from the ontology. Moreover, the structure of ontological instances in often not rigid and can provide multivalued or structured attributes. Finally, when dealing with relational data, the attribute values of records consist quite often in atomic data and the reference to other records through a foreign key is not so frequent. On the contrary, in case of ontological instances expressed by semantic web languages like RDF or OWL, it is very common to

provide attribute values in form of references to other objects, while literal values are less used.

Meaning of the results. Finally, about the meaning of mappings produced as a result of the comparison, we can distinguish between two main approaches. On one side, mappings are interpreted as "same-as" relations, where the two mapped object descriptions are intended to denote the same real object. On the other side, a mapping may be intended just as a generic relation of similarity between object descriptions. In this case, the term *matching* is usually used to stress the role of the comparison more than the interpretation of its results. The term *resolution* instead is more focused on the idea of determining if two descriptions of objects are referred to the same object. More details about the interpretation of mappings meaning is given below.

Other relevant terminology and fields. The problem of data linking and its associated techniques of instance matching and object description comparison is very pervasive also in fields other than the semantic web and for purposes other than the linking of data. For example, the expression *reference reconciliation* has been used with reference to ontological data in [14] and [50]. The problem of comparing different object descriptions has been addressed also in the research community on natural language processing, where linguistic descriptions of objects like names of sentences are compared for disambiguation purposes. In this case we talk about *anaphora resolution* or *named entity disambiguation* [15]. In the database and data mining fields, the problem is relevant also for addressing the problem of *object identification*.

1.3 Mappings and their meaning

The data linking task produces a collections of mappings that are used to define links between object descriptions in different data sources. In this context, a relevant issue is to determine the meaning of these mappings. Usually, the mappings are interpreted as binary relations between two object descriptions that could be considered *identical* in that they refer to the same real object. The notion of identity in itself, however, is ambiguous and is a subject of philosophical discussions. In literature about ontologies and the semantic web, identity has been described as the problem of distinguishing a specific instance of a certain class from other instances [22]. Thus, the identity criterion is linked to the class to which an instance belongs and depends on the assumptions made by the ontology designer. In this way, identity mappings between instances should, in principle, follow the identity criterion associated with the corresponding classes. However, instance matching usually deals with datasets originating from different sources, and the assumptions of their creators may be potentially different. Moreover, the design practices in the semantic web are always not explicitly stated and do not follow a standard approach. In the Linked Data domain, identity is commonly expressed using the standard *owl:sameAs* property. In OWL, two individuals connected by the *owl:sameAs* property are considered identical in the sense that the subject and object of this statement share all their properties. Such interpretation of identity, however, appears too strong in many cases and it is not always compatible with the evaluation of mappings, which is based on similarity. In order to understand the practices of the usage of *owl:sameAs* links by existing repositories, the authors of [23] distinguished five weaker varieties of identity beyond the canonical one. The idea of providing more than a single notion of identity is

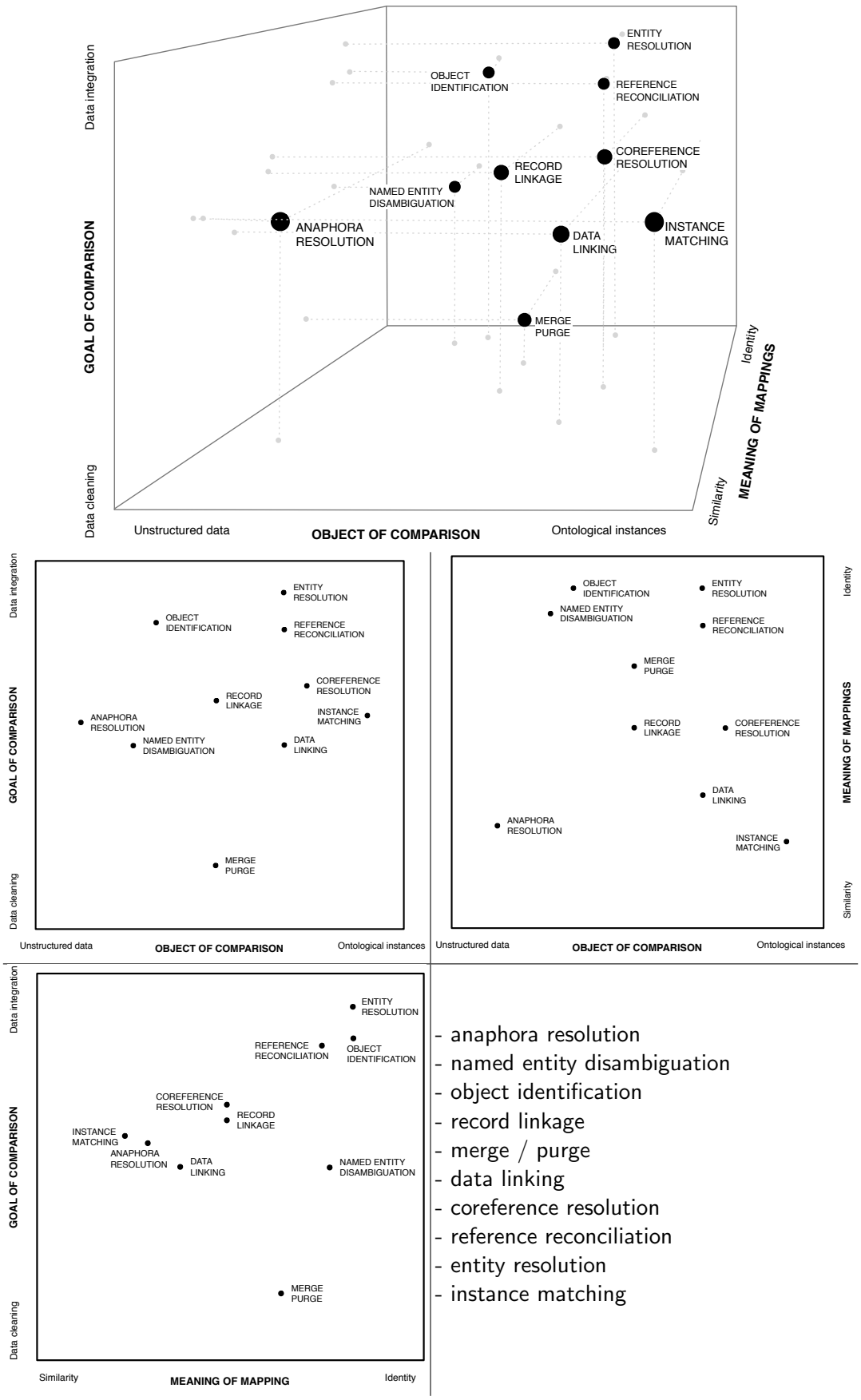

Figure 1.2: A map of the terminology in the field of data linking

also adopted by the UMBEL and the SKOS vocabularies, which provide weaker relations of correspondence such as *umbel:isLike* or the family of "Match" properties of SKOS.

More in general, the problem of having a sound and well founded notion of identity is often ignored in the data linking process. As a consequence, the interpretation of mapping can shift among three very different notions of identity:

1. *Ontological identity:* the idea behind ontological identity is that the real and existing object denoted by two identical object descriptions is the same. In other terms, the properties of the two object descriptions must be the same in order to call them "identical".

2. *Logical identity:* what we refer as logical identity is the idea that two descriptions are identical when they can be substituted one to the other in a logical expression without changing the meaning of the expression.

3. *Formal identity:* in case of formal identity, we refer to the fact that identity is superimposed to the data. This happens for example when some organization sets a unique identifier for something with the goal of providing a standard tool for the identification of the object. For example, in the Datalift platform the IGN and INSEE are providing reference identifiers for geographical and administrative objects on the french territory.

The main problem with the notion of identity in the field of data linking is that the process of matching is always based on the notion of similarity and that the relation between similarity and identity is not clear. This can be shown by the following example: suppose comparing two object descriptions referring to different editions of the same book. Now, the two descriptions are probably similar, but they can be considered identical only if we are interested in the book, seen as a work and not as the edition. But if we look and the manifestation of the book (i.e., the edition) or even to a physical volume in a library, the two descriptions have to be considered different. Thus, the identity is not "internal" to the data and not directly dependent on the relation of similarity, but depends on the reference to something else than data (e.g., in this case the kind of object we are interested in). However, despite the fact that the identity problem is widely recognized and discussed in the research community and that ontologies defining weaker degrees of identity exist (e.g., SKOS[3] or the ontology proposed in [23]), *owl:sameAs* remains the most popular representation of identity links between Semantic Web repositories. this predicate will be in a first step used by Datalift platform to assert similarity between resources. Existing matching systems do not explicitly operate with different interpretations of matching results: instead the choice of interpretation is left to the user of the system.

[3] http://www.w3.org/TR/skos-reference/

2. Instance matching techniques

As was discussed in section 1, the problem of data linking is closely related both to the problem of database record linkage and the problem of ontology schema matching. Data linking systems processing semantic data utilize many techniques originating from both these communities. Thus, in order to determine suitable classification criteria for data linking techniques, we considered both the techniques themselves and existing classification schemas proposed in the literature on database record linkage [15, 62, 32] and schema matching [48, 16]. Based on this analysis, we have chosen the following dimensions for classifying coreference resolution techniques and approaches.

The first dimension we included in our classification schema is *Granularity*, which is also commonly used in the literature on adjacent topics. In the record linkage research there is a distinction between *field matching* and *record matching* techniques, which focus on two different stages of the linkage process [15]. The former concentrate on identifying equivalent values for the same field in two records. This requires resolving the cases of different representation for the same data, such as misspellings, format variations, and synonymy (e.g., "Smith, John" and "J. Smith" or "Yangtze River" vs "Chang Jiang"). The latter focus on deciding whether two records (possibly containing several fields) refer to the same real-world entity. A similar distinction exists in the schema matching research community: e.g., in [48] and [16]. Matching algorithms are classified into *element-level* and *structure-level* ones. When analysing existing data linking techniques with respect to the granularity criterion, we identified three main categories:

1. *Value matching.* Similarly to the field matching task in record linkage, this step focuses on identifying equivalence between property values of instances. In the simplest cases (e.g., when equivalence of two individuals is decided based on equivalence of their labels), no further steps are performed.

2. *Individual matching.* The goal of this stage is to decide whether two individuals represent the same real-world entity or not. Individual matching techniques compare two individuals and utilize the results of the value matching stage applied to properties of these individuals. At this stage, two individuals are considered in separation from all other individuals in two datasets.

3. *Dataset matching.* This step takes into account all individuals in two datasets and tries to construct an optimal alignment between these whole sets of individuals. The techniques handling this stage take as their input the results of the individual matching and further refine them. At this level, mutual impact of pairwise individual matching decisions can be taken into account: e.g., if we know that both datasets do not contain duplicates, then one individual cannot be mapped to two individuals from the other dataset.

As the second classification criterion, we use *Type of Evidence* used by the matching method. Based on this criterion, we distinguish between two categories of methods:

- *Data-level.* These methods rely on information defined at the level of individuals and their property values. In case of ontologies based on description logic, this information corresponds to ontological ABox.

- *Knowledge-level* These methods utilise knowledge defined by the ontological schema (e.g., subsumption relations, property restrictions) as well as in external sources. External sources can include, for example, third-party ontologies as well as linguistic

resources which specify the meaning of terms based on their relations with other terms (such as WordNet [39]).

Finally, we distinguished algorithms based on the *Source of Evidence*:

- *Internal.* These techniques only utilise information contained in the datasets being matched.
- *External.* These techniques exploit information contained in external sources.

To select the techniques for this section, we reviewed different relevant tools and algorithms described in the literature, which included the instance matching task in their process. We use the following criteria for considering a tool or an algorithm as a relevant one:

- It has to operate on semantic data expressed in RDF. Thus, various record linkage tools which assume relational data representation are not included and the methods used in these tools are not mentioned. However, we still consider techniques originated in the database domain if those techniques were later adopted by the tools processing semantic data.
- It has to include the instance matching task in its workflow. We primarily focused on the systems dedicated to data linking (see section 3). However, we also included the techniques used by tools, which performed instance matching as an auxiliary task. In particular, these include generic ontology matching systems which match instance data in addition to schemas (e.g., RiMOM [34] or ASMOV [30]), semantic search tools (e.g., PowerAqua [35]), and identity management servers (OKKAM [7]).

With respect to the data linking workflow shown in Figure 1.1, most systems implement these techniques as a part of the instance matching stage, however, some of the techniques (primarily, dataset matching ones) are sometimes applied during the post-processing & validation step.

2.1 Value matching

Sometimes two individual descriptions contain the same property value expressed in different ways, e.g., because of different formatting conventions of two repositories or the use of synonyms. Resolving these discrepancies and determining that two values are equivalent constitutes the value matching task. As their output, value matching techniques often produce a score denoting the degree of similarity between two values, e.g., that "J. Smith" and "Smith, John" are likely to be two representations of the same name. Although equivalence of values does not necessarily imply equivalence of entities (e.g., there can be many people with the name "John Smith"), value matching techniques serve as building blocks in the instance matching process: their output is aggregated and serves as evidence for making decisions about equivalence of individuals.

2.1.1 Data-level techniques

Data-level value matching techniques primarily involve various *string similarity* metrics widely used in both record linkage and schema matching. These metrics often serve as basic building blocks for more sophisticated methods operating at individual matching and dataset matching levels. External data-level metods are less frequent and include, in particular, standard *keyword search services* such as Google.

Internal approaches To perform value matching, data linking systems widely adopt methods developed for database record linkage, in particular, various string similarity metrics. There are several survey papers reviewing these metrics (e.g., [11], [15], [62]) and we refer the reader to these papers for the detailed description of different similarity functions. Given that there is no single best string similarity measure for all domains, data linking tools usually allow selection among several metrics. In particular, existing tools use the following similarity functions:

- *String equality.* The basic similarity function which returns 1 if two values are equal and 0 otherwise. It has the highest precision, but does not accept slight variations caused by typos or minor formatting differences. Nevertheless, it is used, in particular, by the popular Silk data linking tool as one of the options [61].
- *Edit distance (Levenshtein).* Character-based similarity function which measures the number of primitive change operations needed to transform one string value into another. Examples of systems employing this metrics include [61], [34], and [35].
- *Jaro.* The metric designed to take into account common spelling deviations. It is also employed in the Silk system [61].
- *Jaro-Winkler.* A modification of the Jaro metric adjusted to give a higher score to values which share a common prefix. It is commonly used to compare person names. Examples of usage are given in [59], [51], [61], [43], [27].
- *Monge-Elkan.* This similarity function first involves dividing input strings into a set of substrings and then computing the maximal aggregated similarity between pairs of tokens. In the comparison study described in [11], this metric achieved the best average performance. Among data linking systems, its use is reported in [61] and [43]
- *Soundex.* This is a phonetic similarity function which tries to determine string values which are pronounced similarly despite being different at the character level. The authors of [27] use Soundex similarity as part of their algorithm.
- *Token set similarity.* This metric involves tokenizing the input strings and then comparing resulting sets of tokens using a common set similarity function (such as Jaccard score or overlap distance). It is employed in [61] and [30].
- *I-Sub.* This metric proposed by [57] specially for the ontology matching task calculates similarity between two strings as a function of both their commonalities and their differences. For the purpose of instance matching, this technique is employed in ObjectCoref [26].

We can distinguish two common usage patterns of string similarity metrics depending on the purpose of the system:

- Semi-automated tools which must be pre-configured by the user often include a wide range of similarity functions. The user can select suitable similarity metrics depending on the task at hand. This particularly applies to data linking frameworks designed to deal with RDF data such as Silk [61] or KnoFuss [43].
- The tools which implement an automated workflow usually employ a single metric or an aggregation of several metrics applicable to a wide range of domains. This approach is usually implemented in ontology matching tools which also deal with instance matching

(e.g., RiMOM [34], ASMOV [30], or ILIADS [59]) and semantic search tools (e.g., PowerAqua [35]).

Existing tools usually do not implement the similarity metrics but reuse public API libraries containing these implementations, such as Simmetrics[1] or SecondString[2].

External approaches Standard keyword-based Web search engines provide an additional source of evidence available to matching tools. Normalized Google distance [10] uses the sets of hits returned by Google for two labels being compared as an indication of semantic similarity between these terms. The distance is defined as

$$NGD(x,y) = \frac{max(log(f(x)), log(f(y))) - log(f(x,y))}{log(M) - min(log(f(x)), log(f(y)))},$$

where

- $f(t)$ is the number of Google hits for the search term t,
- $f(x,y)$ is the number of Google hits for the tuple of search terms x and y,
- M is a total number of pages indexed by Google.

This distance measure is adopted for the generic ontology matching task in [19]. One disadvantage of this metric is the time cost of the Web search operations, which complicates its use for data linking tasks potentially involving large number of value comparisons.

For the traditional interlinking method will check every property of the instance. It's not applicable for large data sets. [56] designs an index-based selecting algorithm in order to reduce the comparison set. They find the key from properties to select instances. It produces the key by computing on single datatype properties. If no single datatype property can be the key, the combination of datatype property is considered to be candidate keys. Their method can be applied to any structured data.

One potential drawback of his method might be as follows. This method will consider all datatype properties of all concept in a RDF data set. But not all property of source instance have parallel property of instance from target RDF data set. That is to say, such key could not be used to distinguish candidate matching instances for there is no benchmark from another RDF data set to be referred to for picking out which is the most matching one.

[25] considers not only key properties to find similar RDF data, but also mining frequently linked properties to identify equivalent data pairs. It uses machine learning techniques to enlarge iteratively the property set that could be treated as the key to identify similar instances.

2.1.2 Knowledge-level techniques

Knowledge-level techniques try to discover similarity between property values based on some semantic relations between them. In order to obtain this information, the tools involve external knowledge sources. These sources can be classified into two main categories:

- *Linguistic resources.*
- *Formal domain models.*

[1] http://staffwww.dcs.shef.ac.uk/people/S.Chapman/simmetrics.html
[2] http://secondstring.sourceforge.net/

Linguistic resources provide relations between words used in the property values. WordNet [39] is the most commonly used thesaurus which contains such relations as synonymy and hyponymy, which are used directly for value matching. In particular, it is employed in [8], [21], [52], [35]. Similarity between terms is computed using distance measures based on the length of path between two entities in the graph. For example, the term affinity function which is used in HMatch [8] is defined as

$$A(t,t') = \begin{cases} max_{i=1..k} W_{t \rightarrow_i^i t'} & if k \geq 1 \\ 0 & otherwise \end{cases}$$

where k is the number of paths between t and t' in the graph, $t \rightarrow_i^i t'$ is the i-th path of length n, $W_{t \rightarrow_i^i t'} = W_{1_{tr}} \cdot W_{2_{tr}} \cdots W_{n_{tr}}$ is the aggregated weight of the i-th path which equals the product of weights associated with each edge in the path. Weights of edges are set depending on the type of the corresponding relation (hyponymy, synonymy, etc.).

In a specific case where the datasets contain field values expressed in different languages, multi-lingual dictionaries are used in addition to pre-process values and translate them into a default language (e.g., English). Translation procedures are applied by the OKKAM server [7].

More specialised relations between terms are represented in publicly available ontologies. High-level vocabularies such as SUMO[3], SKOS[4], OpenCYC[5] contain terms from a wide range of domains. In particular, taxonomic distance based on SKOS is used in RDF-AI [52].

The approaches based on formal domain models, however, are less commonly used for the instance matching task than in the schema matching community. This is due to the fact that these models mainly cover common terms, which correspond to classes in the ontologies, rather than entity names, which denote labels of individuals.

2.2 Individual matching

Establishing links between individuals which refer to the same real-world entities constitutes the main goal of the matching process. Individuals are described using their properties as well as relations to other individuals in the dataset. Individual matching techniques decide whether two individuals taken from different datasets should be linked or not using descriptions of these individuals as evidence.

2.2.1 Data-level techniques

Data-level individual matching normally involves aggregating the results of value matching methods over the values of properties belonging to two individuals being compared. Different *attribute-based similarity functions* have been proposed to perform such aggregation. In addition, existing *external mapping sets* are utilised to infer mappings between individuals in the context of Linked Data cloud.

Internal approaches The classical approach to individual matching includes aggregating the results produced by value matching techniques. This model for solving the record linkage problem was proposed in a seminal paper by Fellegi & Sunter [17]. The model assumes that there exist two lists of records (A and B) describing some real-world entities. The task of

[3] http://www.ontologyportal.org/
[4] http://www.w3.org/2004/02/skos
[5] http://www.opencyc.org

record linkage is to classify each pair of records from the set $A \times B$ into two sets: M (set of matched pairs) and U (set of non-matched pairs). Given that each real-world entity a or b is described using records $\alpha(a)$ and $\beta(b)$, the classification decision is based on a comparison of two records expressed as a vector function $\gamma(\alpha(a), \beta(b)) = \gamma(a, b)$ (distance function). The authors propose a probabilistic model introducing conditional probabilities $m(\gamma) = P(\gamma[\alpha(a), \beta(b)]|(a, b) \in M) = \sum_{(a,b)\in M} P(\gamma[\alpha(a), \beta(b)]) \cdot P((a, b)|M)$ and $u(\gamma)$ (similar for $(a, b) \in U$). The obtained value $m(\gamma)/u(\gamma)$ is used to make a decision about the equivalence of two entities. Possible decisions are denoted as A_1 - positive link, A_2 - possible (uncertain) link, and A_3 - positive non-link. The decision is chosen by comparing the value $m(\gamma)/u(\gamma)$ with thresholds T_μ, such that if $m(\gamma)/u(\gamma) > T_\mu$ then $P(A_1|U) < \mu$, and T_λ, such as if $m(\gamma)/u(\gamma) < T_\lambda$ then $P(A_3|M) < \lambda$, where μ and λ are desired error levels. The challenges in this classical model include calculating $m(\gamma)$ and $u(\gamma)$, threshold values T_μ and T_λ, and the form of the comparison function γ. In the original method proposed in [17], components of the vector $\gamma = (\gamma_1, \gamma_2, ..., \gamma_K)$ are the results of pairwise field values comparison produced by value matching techniques.

This model served as the base for the majority of algorithms developed in the record linkage domain (see [15] for a survey) and was re-used in several tools developed in the Semantic Web community. The aggregation methods adopted by these tools include the following:

- *Weighted average.* $\gamma = \frac{\Sigma_{i=1}^{K} w_i \gamma_i}{K}$, where w_i denotes the weight of the ith element of the attribute comparison vector.
- *Picking maximal (minimal) value.* $\gamma = max_{i=1}^{K}(\gamma_i)$
- *Euclidean distance.* $\gamma = \sqrt{\Sigma_{i=1}^{K} \gamma_i^2}$
- *Weighted product.* $\gamma = \Pi_{i=1}^{K} w_i \gamma_i$
- *Group linkage.* $\gamma = \frac{\Sigma_{i=1}^{K} s_i}{|Q|+|C|-|M|}$, where $s_i = \{\gamma_i$, if $\gamma_i > t$, 0 otherwise$\}$, $\{Q, C\}$ are the sets of predicates of two individuals, and M is the number of elements for which $\gamma_i > t$
- *Log-based tempered geometric mean.* $\gamma = exp^{\frac{\sum_{i=1}^{t} w_i log(\gamma_i+\epsilon)}{\sum_{j=1}^{t} w_j}} - \epsilon$.

In particular, the Silk system [60] assumes a supervised matching scenario where the user selects an aggregation approach (weighted average, max(min), Euclidean distance, or weighted product) for her task. Similarly, the the ODDLinker system [24] used to discover interlinks for the LinkedMDB repository represented RDF individuals as relational tuples and employed aggregated attribute-based similarity to decide on equivalence of individuals. In contrast, the OKKAM Entity Naming Service [7] is targeted at the unsupervised matching scenario in an open environment and employs automatic selection of an aggregation function. In order to achieve high precision, more involved functions (such as group linkage or geometric mean) are utilized [28].

External approaches While internal data-level individual matching techniques are largely adopted from traditional record linkage research, external approaches often utilize the features specific for the Web environment, and, in particular, Semantic Web. Sets of pre-existing identity links between individuals declared in distributed Linked Data repositories constitute one such feature. Using the transitivity of the *owl:sameAs* property, these identity

links can be aggregated to infer the equivalence between individuals connected via a chain of *owl:sameAs* links. This approach is directly exploited by the authors of *sameas.org* portal [6], which aggregates equivalent Linked Data resources into "coreference bundles" using both explicit *owl:sameAs* links and inferred ones. The idMesh system [13] employs this method as its first step to construct the graphs connecting potentially equivalent RDF individuals, which it later refines using the graph analysis techniques. The ObjectCoref system [26] also uses existing sets of individuals linked via the *owl:sameAs* property to bootstrap learning of discriminative properties which are later used to infer new equivalence mappings.

2.2.2 Knowledge-level techniques

The main source of knowledge which can be utilised by individual matching algorithms are *ontological restrictions* defined by domain ontologies used to structure the data in the matched repositories. In cases where the repositories are structured using different schema ontologies, *ontology mappings* obtained from external sources can be exploited to establish correspondences between schema terms and align the structure of individual descriptions in both repositories.

Internal approaches The main sources of information used by internal knowledge-level approaches are the domain ontologies which define the structure of individuals in the input repositories. Logical axioms defined in the ontologies can provide both "positive" evidence from which new equivalence mappings can be inferred as well as "negative" evidence which can be used to filter out spurious mappings. Because of this, methods based on the use of ontological constraints are often applied to refine an initial set of candidate equivalence mappings obtained using other methods (such as attribute-based similarity).

In particular, the following ontological constructs are valuable for the individual matching task and are often directly utilised by matching systems:

- *owl:FunctionalProperty* (*owl:InverseFunctionalProperty*). These properties essentially define the key attributes for individuals similarly to database keys. Equivalent values of inverse functional properties imply the equivalence of the subjects of properties. Functional properties, in a similar way, allow equivalence of objects to be inferred if they are connected to the same subject. Functionality provides a strong restriction which can be useful both for inferring the new *sameAs* links between instances and for detecting spurious existing links. If two individuals were initially considered as equivalent and connected by a *owl:sameAs* mapping but have two different values for the same functional property, this can indicate that the mapping was incorrect.

- *owl:maxCardinality*. Similarly to the functionality restriction, cardinality violation can also point out to an incorrect mapping (although it does not directly).

- *owl:disjointWith*. As another type of negative evidence, disjointness between classes can be used to reject candidate mappings if they connect instances belonging to different classes.

Among the existing systems, Sindice [58] implements a method for instance matching using explicitly defined inverse functional properties as positive evidence to infer equivalence between instances. The L2R algorithm [50] utilises all relevant ontological schema constraints mentioned above and uses them as both positive and negative evidence. ObjectCoref [26]

[6] http://www.sameas.org

also employs ontological restrictions together with explicit *owl:sameAs* links to create the initial "kernel" mappings. These "kernel" mappings are used as a training set to learn discriminative properties, which in turn help to infer more individual mappings.

While logical restrictions are valuable in the data linking task, it is difficult to exploit them if the datasets are structured using different schema ontologies: precise translation of ontological axioms is problematic for automatic ontology matching systems. ILIADS [59] is an ontology matching system which performs schema- and data-level matching in iterations. After each data-level matching iteration, restrictions defined in both ontologies being matched are used to check the validity of obtained mappings between instances. Then, at the schema-level iteration, these mappings are used as evidence to infer mappings between classes (by using set similarity metrics over the sets of their instances).

External approaches External knowledge sources are particularly valuable in the data linking tasks involving repositories structured using different ontologies. Such tasks are common in the Linked Data environment, and existing *schema mappings* and *schema restrictions* represent a commonly used knowledge resource.

In [42] schema mappings are inferred using existing *owl:sameAs* links between Linked Data repositories. In case where instances of two repositories are not linked to each other directly, but there exist connections to the same external repositories, such links are used as evidence for instance-based ontology matching. For example, both instances *movie:music_contributor/4026* from LinkedMDB[7] and *dbpedia:Ennio_Morricone* from DBPedia[8] are connected to *music:artist-a16e47f5-aa54-47fe-87e4-bb8af91a9fdd* from MusicBrainz[9]. Aggregating sets of such composite mappings and using set similarity functions, the system can derive mappings between corresponding classes *movie:music_contributor* and *dbpedia:MusicalArtist*. Schema-level mappings obtained in this way present an additional input to the KnoFuss system [43]. They are used to determine subsets of two repositories which are likely to contain identical instances and to perform instance matching in the same way as in a single-ontology scenario.

An extension to the CIDER ontology matching system [21] described in [20] uses ontological context to disambiguate ontological entities (both classes and individuals), e.g., to disambiguate the country name "Turkey" from the name of the bird. Ontological context consists of all neighbouring ontological terms and is obtained by querying the Watson ontology search service[10]. This ontological context is then used to cluster different senses of the same term available on the Semantic Web as a whole and to assign the ontological entities being compared into appropriate clusters. This approach, however, is primarily targeted at the schema level and has a limited applicability to instance matching. For instance, it is difficult to distinguish between two individuals belonging to the same class as there would be little difference in their ontological context.

2.3 Dataset matching

Data individuals inside repositories are usually interrelated with other individuals. Thus, a decision about matching a pair of individuals can influence the confidence in matching another pair of individuals. For example, deciding that two publication references in two citation datasets refer to the same paper also increases the confidence in matching the references to their authors. To capture and utilise such interdependencies, one must analyse the

[7] http://www.linkedmdb.org/
[8] http://dbpedia.org
[9] http://dbtune.org/musicbrainz/
[10] http://watson.kmi.open.ac.uk/

whole set of potential mappings between two datasets rather than each pair of individuals in isolation. Dataset matching techniques require a set of candidate mappings to be provided as input and thus are usually applied after the individual matching stage.

2.3.1 Data-level techniques

Data-level dataset matching techniques involve the analysis of graphs formed by both relations between individuals within repositories and identity mappings across repositories. Internal matching techniques perform this on the basis of two datasets being matched while external ones involve information from third-party sources, in particular, Linked Data repositories containing links to the input datasets. To reason about the data statements and individual mappings in these graphs, various *belief propagation frameworks* are used.

Internal approaches Existing systems commonly employ various similarity propagation techniques. These techniques exploit the principle that the similarity between two nodes in the graph depend on the similarity between their adjacent nodes. A generic graph matching algorithm called similarity flooding [38] is used both in schema- and data-level ontology matching (in particular, in the RiMOM system [34]). Similarity flooding includes the following stages:

- Transforming datasets into a directed graph in which pairs of entities (potential matches) correspond to nodes. Edges between two nodes exist in both directions if in both datasets there is a relation between the corresponding entities.
- Assigning weights to the edges. Usually, $w_{ij} = 1/n$, where w_{ij} is the weight of the edge from the node i to the node j and n is the number of outgoing edges of the source node i.
- Assigning initial similarity σ^0 to nodes. The value of σ^0 is usually taken from the output of the individual matching stage.
- Computing σ^{i+1} using a weighted linear aggregation function. The default function is defined as
$$\sigma^{i+1}(x, x') = \sigma^0(x, x') + \sum_{e_p} \sigma^i(y, y') \times w(e_p),$$
where $\sigma^i(x, x')$ is the similarity value for the node representing the mapping between entities x and x', $e_p =<< y, y' >, p, < x, x' >>\in G$ is an edge from the node $< y, y' >$ to the node $< x, x' >$ with the label p, and $w(e_p)$ is the weight of this edge. Resulting values σ^{i+1} are normalised after each iteration.
- The procedure stops if no similarity value changes more than a particular threshold ϵ or after a pre-defined number of iterations.

In [14] another propagation algorithm is used where the graph includes not only individual matching nodes but value matching nodes as well. Nodes corresponding to pairs of individuals are connected by edges to nodes corresponding to pairs of their property values: e.g., a node representing a potential mapping between two publications {a_1, a_2} has edges from the node representing the similarity between titles {"Title_1", "Title_2"} and publication years {1978, 1979}, expressing the dependency of the individual similarity from value similarities. In this way, the propagation algorithm combines the value matching, individual matching, and dataset matching stages in a single workflow. At each iteration, the algorithm

makes a decision about whether a pair of individuals should be merged. Each individual in a merged pair is assumed to contain all the properties of its counterpart, and the graph is updated accordingly. Impact factors which specify how a similarity of node influences its neighbours are chosen on the basis of the type of property corresponding to the edge and the class of instances included in the node. These parameters are set by human users.

In a similar way, the RDF-AI system [52] implements an algorithm which propagates similarities from value similarity nodes to resource similarity ones. This algorithm does not assume the use of the same schema ontology by two datasets. The algorithm first selects pairs of matching property values. At each iteration, a pair of values with the highest similarity is selected as potentially matching and other candidate pairs containing the same values are discarded. Then, similarities are propagated from value matching nodes to individual matching ones. The aggregation function chosen by the authors calculates a new resource similarity as average of the neighbouring value similarity nodes.

The approach presented in [27] is based on the interpretation of similarities as Bayesian probabilities. Accordingly, Bayesian networks [47] are used as the framework for belief propagation in graphs. The propagation is performed by the standard message passing algorithm described in [47].

External approaches External data-level dataset matching approaches aggregate mutual impact of individual matching decisions in a set of several repositories rather than only two at a time. This method is particularly relevant in the Linked Data environment which represents a network of distributed interconnected repositories. These interconnections are usually created by applying data linking to a pair of repositories. In the idMesh system [13], these sets of mappings between individuals are combined over the whole network of data repositories and refined by analysing mutual impact of these mappings. In this way, idMesh can be considered a meta-level data linking system. idMesh builds graphs based only on equivalence and non-equivalence relations between entities and uses factor-graph message passing to compute marginal probabilities. As mentioned in section 2.2.2, ObjectCoref [26] uses a set of mappings between individuals collected from the whole network of distributed semantic repositories as a training set to learn discriminative data patterns.

2.3.2 Knowledge-level techniques

Dataset matching techniques often rely on relations between individuals as evidence. These relations can either be declared or assumed: e.g., having a hypothesis that all individuals within a repository are different from each other. To decide how a specific relation impacts a matching decision, data-level techniques utilise various statistic-based heuristics (e.g., number of incoming/outgoing properties in similarity flooding). One disadvantage of the purely data-level techniques is ignoring explicit definitions of relations provided by the domain ontologies. Knowledge-level methods aim at improving the dataset matching results by utilising this information. These methods usually enrich the *belief propagation algorithms* operating purely at the data level. However, one of the proposed approaches also implements dataset matching as a standard *linear programming* problem.

Internal approaches Internal knowledge-level dataset matching techniques utilise information defined in the domain ontologies which describe the structure of data in the repositories.

LN2R system [51] employs similarity propagation as a part of the "numerical" matching algorithm N2R which is combined with the "logical" one (L2R). Unlike the algorithms de-

scribed in section 2.3.1, this procedure is schema-aware. It employs the aggregation function

$$\sigma^{i+1}(x, x') = max(\sigma_f^{i+1}(x, x'), \sigma_{nf}^{i+1}(x, x')),$$

where $\sigma_f^{i+1}(x, x')$ is the aggregated similarity function over the edges representing functional properties and $\sigma_{nf}^{i+1}(x, x')$ corresponds to non-functional ones. Similar to [38], the impact of non-functional attributes is reduced proportionally to the number of edges corresponding to the same property.

In [44] the results of value matching, ontological axioms, and relations between individuals are combined together using valuation networks [55]. Valuation networks are graphs containing two kinds of nodes. Variable nodes correspond to the confidence degrees of individual matching decisions and data statements[11]. Valuation nodes represent the rules which determine the valid combinations of states of the neighbor variable nodes and correspond to ontological axioms and weak relations. Confidence degrees are interpreted as Dempster-Shafer belief distributions [54].

Unlike the majority of dataset matching approaches which are based on various algorithms for belief propagation on the graph, the authors of [46] implement the procedure for global optimisation of the whole set of mappings. They use class subsumption relations in addition to property restrictions. Their method measures the impact of non-strict ontological relations relevant for a particular mapping between instances. For example, having two potential instance mappings (a, b_1) and (a, b_2) and the concept C defined in the ontology such that $a, b_1 \in C$, but $b_2 \notin C$, the first mapping is preferred because it is better "compatible" with the domain ontology. They define the A-Box similarity measure which quantifies the degree of similarity between two ontological A-Boxes described in terms of the same ontological T-Box. This similarity metrics relies on the value of the overlap function $overlap_{\mathcal{T}}(\mathcal{A}_1, \mathcal{A}_2, \mathcal{M})$ between A-Boxes $\mathcal{A}_1$ and $\mathcal{A}_2$ induced by an instance alignment $\mathcal{M}$. The weighted overlap function measures the aggregated "compatibility" of all pairwise mappings included in $\mathcal{M}$.

$$overlap_{\mathcal{T}}^{w}(\mathcal{A}_1, \mathcal{A}_2, \mathcal{M}) := overlap_c + overlap_p + overlap_{\neg c} + overlap_{\neg p},$$

where

- $overlap_c = \sum_{(<a,b>)} \sum_{(C(a) \wedge C(b))} \sigma(a, b)$, where $< a, b > \in \mathcal{M}$ and $C \in \mathcal{T}$ - aggregated confidences $(\sigma(a, b))$ of mappings connecting instances belonging to the same class,
- $overlap_p = \sum_{(<a,b>,<a',b'>)} \sum_{(P(a,a') \wedge P(b,b'))} \frac{\sigma(a,b)+\sigma(a',b')}{2}$, where $< a, b >, < a', b' > \in \mathcal{M}$ and $P \in \mathcal{T}$ - impact of mappings between pairs of individuals connected by the same property in both repositories,
- $overlap_{\neg c} = \sum_{(<a,b>)} \sum_{(\neg C(a) \wedge \neg C(b))} \sigma(a, b)$ and
- $overlap_{\neg p} = \sum_{(<a,b>,<a',b'>)} \sum_{(P(a,a') \wedge P(b,b'))} \frac{\sigma(a,b)+\sigma(a',b')}{2}$, where $< a, b >, < a', b' > \in \mathcal{M}$, $C \in \mathcal{T}$, and $P \in \mathcal{T}$ - impact of corresponding negated concept and property assertions.

[11] Since the approach is aimed at processing annotations extracted from text, data statements are not considered 100% reliable.

At the next step, weighted A-Box similarity has to be maximised, in other words the value of $arg \max_{\mathcal{M} \in \mathbb{M}} (\mathcal{A}_1, \mathcal{A}_2, \mathcal{M})$ has to be determined. The authors propose two alternative approaches to achieve this. The first one involves transforming the problem into an integer linear programming problem [53] and solving it using standard methods. However, the disadvantage of these methods is their computational complexity which makes it difficult to apply them to large-scale repositories. In order to reduce the computational cost, the authors included an alternative approach which involves applying a generic algorithm for inexact graph matching [12]. This algorithm reduces execution time with some loss of precision.

External approaches External knowledge sources can contribute with additional information of the same kind as utilised by internal techniques, namely ontological restrictions, subsumption and equivalence relations between schema terms, information about properties valuable for determining identity, etc. Relevant knowledge sources include, among others:

- Schema-level mappings obtained using third-party services. Such mappings can be used to translate ontological restrictions from the terminology of one ontology into another.
- Existing interlinked data repositories. These repositories can be used to infer schema-level mappings using instance-based ontology matching (as in [42]) and to mine common data patterns such as discriminative properties for matching (as in [26]).

2.4 Time-Efficient similarity algorithms

Linking a large amount of entities is a challenging task which has been the focus of several works [41]. This is an important problem as the size of datasets to be interlinked can be very large (eg. DBPedia) and thus require a consequent amount of time. Since comparing every pair of instances is time consuming, some blocking methods and candidate selection algorithms have been proposed to prune the search space. Blocking-based techniques (i.e. standard blocking, sorted-neighborhood, bi-gram indexing, canopy clustering and adaptive blocking) propose different ways to form blocks of entities sharing an identical or approximate key [4]. For example, the key in Standard Blocking can be composed from the entity attributes. As for the bigram indexing, it convert the key values into a list of bigrams and build a sub-lists of all possible permutations using a threshold. The resulting bigram lists are sorted and inserted into an inverted index, which will be used to retrieve the corresponding entities numbers in a block. The performance of such techniques is very sensitive to the choice of the key where the entity pairs of true matches have to be in the same block.

In addition, some inverted index based algorithms propose a time-efficient string similarity such as the AllPairs, PPJoin and PPjoin+. The AllPairs [5] provides certain optimization strategies to inverted index-based approach. Indeed, rather than exploiting the similarity threshold to reduce candidate pairs, it exploits it to reduce overhead such as index construction and inverted list scanning. The PPJoin [63] combines positional filtering with prefix filtering method. To cope with the presence of stop words, tokens in each record are sorted according to document frequency ordering. Then, using positional information, PPJoin computes the upper bound of the overlap between two entities x and y. This maximum possible overlap is a sum of the overlap computed on the left partitions using the prefix filtering algorithm, and the minimum number of unseen tokens on the right partitions of x and y. To prune more candidates, the algorithm PPJoin+ [63] propose to integrate a suffix filtering method. The idea is to choose an arbitrary token from the suffix of an entity x to create

the right and left partitions (x_r, x_l). As tokens are sorted by a global ordering, $x_l(x_r)$ shares no common tokens with $y_r(y_l)$. Thus, the lower bound Hamming distance of the suffixes is the sum of Hamming distance of right and left partitions. This algorithm can prune some candidates whose the lower bound Hamming distance is larger than an allowable threshold H_{max} which depends on similarity threshold.

To deal with numeric values, a time efficient method (HYPPO) has been proposed in [41]. It stands for HYpersphere aPPrOximation algorithm and operates in a n-dimensional metric space Ω. In an orthogonal space, a distance metric can be decomposed into the combination of functions $\varphi_{i,i\in\{1...n\}}$ which operates on one dimension. Assuming that the dimensions of Ω are independent, HYPPO holds that $\varphi_i(x, w) \leq \delta(x, w)$ where $\delta(x, w)$ is the distance metric between two n-dimensional points x and w, and θ is a distance threshold. Based on this inequality, the hypersphere $H(w, \theta) = \{x \in \Omega : \delta(x, w) \leq \theta\}$ is a subset of the hypercube $V(w, \theta) = \{x \in \Omega : \forall i, \varphi_i(x_i, w_i) \leq \theta\}$. HYPPO tiles Ω into hypercubes using a granularity parameter and approximate $H(w, \theta)$ by several hypercubes. It discards all elements which are not in $V(w, \theta)$ and it is guaranteed not to lose any link while most blocking techniques are not lossless.

2.5 Summary

As we can see from the analysis of the state of the art on data linking for the Semantic Web, existing systems to a large extent reuse the techniques developed for database record linkage and schema matching. However, the data linking task possesses several features which distinguish it from both these adjacent research directions. These features are exploited by individual matching and dataset matching methods specifically aimed at the Semantic Web environment. Such specific features include:

- Rich structure and additional expressivity provided by ontological schema definition languages (especially OWL) in comparison with relational database schemas. Advanced schema constructs (e.g., class hierarchy, disjointness relations, etc.) can be used to refine the mappings produced by individual matching and dataset matching methods.

- Availability of large volumes of publicly available data and existing mappings between them. In the Linked Data cloud, data linking systems can consider a whole network of repositories instead of only two as in classical record linkage and ontology matching scenarios.

3. Data linking systems

Data linking systems implement a number of the techniques identified in Section 2 in order to interlink Web data described in RDF.

In the following analysis, we study 11 systems performing both automated and semi-automated data linking. While many ontology matching systems are also able match instance sets or use instance matching techniques in the ontology matching process (see [8, 34, 21, 30, 46, 59]), we focus here on systems whose primary focus is to perform data linking. A few other systems dealing with Web data use data linking techniques while not focusing on linking data and are thus not included in this section (see [35, 7, 58]). More generally, many ad-hoc matchers are designed when interlinking a specific dataset on the Web of data.

Semi-automated data linking systems typically use configuration files that need to be adapted for each pair of datasets. Parameters such as matching techniques, properties to be compared and thresholds need to be entered by the user. This manual input is in most cases needed if a high link quality has to be reached. Automated matching with high quality links can be achieved if the domain of the tool is well defined, such as for LD mapper [49] for the music domain.

Each tool is described below, then in Section 3.12. Tables 3.1, 3.2, and 3.3 specify which of the techniques we present in Section 2.

3.1 LN2R

LN2R [51] is an unsupervised instance matching system. It uses two approaches: one logical (L2R) and one numerical (N2R). L2R exploits the axioms of the ontologies describing instances. Both datasets should be described using the same set of ontologies. Functional and inverse functional properties, as well as disjoint classes axioms are considered. Matches and non-matches found by L2R are then used as input for the numerical similarity method N2R. N2R uses equations modelling dependencies between similarities of entities. This capture the intuition that if two pairs of instances are related in two datasets and two of these instances are similar across datasets, then the two other instances are likely to be similar as well. Similarity between attributes is computed using an external thesaurus or string similarity algorithms. Then an iterative algorithm compute the similarity of instances based on the attributes similarities and the dependencies equations.

3.2 ObjectCoref

ObjectCoref [26] aims at building a searchable repository of identity links between resources on the Web of data. The data linking system is based on a self-training network, a semi-supervised learning framework. The system thus needs a training set before being able to interlink two datasets. ObjectCoref uses ontological axioms to further discover equivalences between resources: existing owl:sameAs links, functional and inverse functional, as well as cardinality restrictions. A discriminating factor is then computed for pairs of properties belonging to matching resources in the training set. The system then performs an analysis of property-value pairs in order to ascertain which properties have a similar value for resources in the training set. In fact the system performs on the fly matching of properties used in the two datasets. The matching properties are then used to link the two datasets in an iterative process.

3.3 Okkam

Okkam [28] proposes an architecture based on the use of distributed servers maintaining sets of equivalent resources. They are named Entity Name Servers (ENS). Each equivalent resource set is assigned a identifier. An ENS store entity descriptions on the form of key/property values. New entities are added based on a matching algorithm constructing a similarity measure between the candidate resource and the ENS entity. The similarity measure uses a string matching algorithm between the key/property pairs. The similarity measure is then weighted according to the likelihood of the key to indicate a name for the entity. A small vocabulary of naming properties is thus maintained in the system. Finally similarities are aggregated by computing the sum of maximal similarities between the features of the two entities.

3.4 RKB-CRS

The co-reference resolution system of RKB [18, 29] consists of resource equivalence lists. These lists are constructed using ad-hoc Java code on the specific conference/university domain. Domain heuristics are provided such as co-authorship analysis. New code needs to be written for each dataset. It consists in selecting resources to be matched and performs string similarity matching on relevant attributes.

3.5 LD-mapper

LD mapper [49] is a Prolog based dataset interlinking tool. The tool is based on a similarity aggregation algorithm taking into account the similarity of neighbor resources. The tool requires little user configuration and has been tested on music related datasets. The current implementation works with the Music Ontology, but the algorithm can be used on datasets working with any ontology.

3.6 Silk

Silk [61] is a framework and tool for interlinking datasets and maintaining the links. It consists of a tool and a link specification language: the Silk Link Specification Language. Before matching two datasets, the user specifies entities to link in a LSL file. The tool uses various string matching methods, but also numeric equality, date equality, taxonomical distance similarity, and sets similarity measure. All these similarity measures are parameterized by the user using a specific format. Preprocessing transformations can be specified by the user in order to improve the quality of the matching. Matching algorithms can be combined using a set of operators (MAX, MIN, AVG). Also, literals can be transformed before the comparison by specifying a transformation function, concatenating or splitting resources. Silk takes as input two datasets by specifying SPARQL endpoints. It is able to output sameAs triples or any other predicate between the matched entities. Silk was tested on diverse datasets available on its project Web page.[1]

[1] `http://www4.wiwiss.fu-berlin.de/bizer/silk/spec/`

3.7 LIMES

LIMES [40] is a semi-automatic interlinking tool that can be configured using a XML specification format. Its originality lies in the usage of the properties of metric spaces in order to optimize the use of matching techniques and reduce the number of comparisons needed to match two datasets. The tool needs user input formatted using a link specification language. The tool is available online as a Web service[2] a graphical user interface to a Web service is also available at [3].

3.8 KnoFuss

The KnoFuss architecture [43] is designed for the fusion of heterogeneous knowledge sources. One particularity of KnoFuss is its ability to match datasets described under heterogeneous ontologies. The matching process is driven by instanciating an ontology specifying resources to be matched, and the adequate matching techniques to use for these resources. For each type of resource to be matched, an *application context* is defined, specifying a SPARQL query for this type of resource. A variety of string similarity algorithms are available. When datasets described by heterogeneous ontologies are used, it is possible to specify an ontology alignment in the alignment format[4], thus allowing to use any matcher outputting this format. The tool is primarily meant to perform fusion of two input dataset. The input fusion ontologies thus also specify how should the resources be merged. A post-processing step is performed to verify and enforce the consistency of the new dataset resulting from the fusion operation with regards to the ontologies. The tool works with local copies of the datasets and is implemented in Java.

3.9 RDF-AI

RDF-AI [52] is an architecture and prototype implementation for datasets matching and fusion. The tool generates an alignment that can be further used to either merge the two matched datasets or produce a set of links containing the owl:sameAs triples. The system takes as input XML files specifying the preprocessing parameters: name reordering, property strings translation, datasets ontological structure, and matching techniques for each kind of resource. The datasets structure and the resources to be matched are described in two files. This descriptions in fact corresponds to a small ontology containing only resources of interest and the properties to be used in the matching process. Another configuration file describes post-processing parameters such as the threshold for generating links, as well the fusion parameters in case of a fusion. The tool works with local copies of the datasets and is implemented in Java.

3.10 Zhishi.links

Zhishi.links [45] is a general purpose data linking tool using a distributed framework to reduce the elapsed time when matching large datasets. Resources are therefore indexed before similarity calculations are performed. Two similarity comparisons are available, one generic based on entities names (RDFS and SKOS lablels, aliases) and based on geographical

[2] `http://aksw.org/Projects/limes`
[3] `http://limes.aksw.org/`
[4] `http://alignapi.gforge.inria.fr/format.html`

location. Then a semantic matching technique is used to increase the similarity of instances sharing same property-value pairs. Candidates are finally sorted by their similarity value score, using an eventual new computation on their default label to disambiguate candidates having the same score. The system also uses abbreviations list for persons, locations and organizations (ie "Jr" for junior).

3.11 Serimi

Serimi [2] interlinking tool use a two phases approach, selection and disambiguation. In a first phase, traditional information retrieval strategies with high recall but low precision are applied to select candidate resource to be linked. More precisely, entity labels of the source dataset are used to search for entities in the target dataset. This step is to reduce the comparison space for the next step. In a second phase, candidates matches sharing same labels are disambiguated using an algorithm that identifies which of the candidates is more likely to be the correct one. This technique is based on an analysis of resources descriptions, by identifying property sets shared by instances. *Classes of interest* are thus formed. Instances of the same class in the source dataset are finally linked to instances of the same class of interest in the target dataset. The Serimi method could be applied in various domains, for is is claimed that it could find matching instances regardless of prior knowledge on context and schema. But this interlinking method has its risk. The string matching used in first step has high recall and low precision, so the selected instance candidate sets may ignore certain similar instances. For the comparison in second step is based on the selected triples coming from the first step, so the interlinking result might ignore false negative matches.

3.12 Summary

As we can see in tables 3.1, 3.2, and 3.3, varying data linking techniques are implemented by the studied systems.

It seems obvious that internal data level techniques are implemented in every system for value matching as they are the basic operation for comparing instances values. Only two systems use translation services to translate strings before comparing them. This can surely be explained by the fact a large majority of the data available on the Web of data is in English. External knowledge sources are commonly used to compute similarity between instances values. Most tools use similarity aggregation in order to combine similarity over many attributes of the matched instances, but few tools make use of existing links (data level, external techniques). As the density of links on the Web of data increase, reusing existing links or computing transitivity closures might be useful. It is however crucial to consider links quality in order to prevent the propagation of incorrect links. Three tools only consider the ontology axioms when computing the similarity of two instances. One reason behind this is that few vocabularies on the Web of data actually have these axioms. As more work will be done on increasing vocabularies quality, data linking using ontological axioms will become more and more important. Dataset matching techniques are the least used ones. Indeed they rely on the previous set of techniques and thus are the most advanced ones. One tool only makes use of third-party schema mappings. This can be seen as very low given that reconciling schemas greatly facilitates the matching task. Semi automated tools actually solve this problem by implicitly letting the user to specify the schema alignment when configuring the tool for a specific matching task.

Table 3.1: Systems value matching techniques

	Value Matching		
	Data level		Knowledge Level
System	Internal	External	
LN2R	string matching		Wordnet synonyms dictionnary
ObjectCoref	string matching		
OKKAM ENS	string matching	translation service	entity names vocab.
RKB-CRS	string matching		
LD Mapper	string lookup search		
Silk	string matching numerical similarity		taxonomic distance
LIMES	string matching on metric spaces		
KnoFuss	string matching		
RDF-AI	fuzzy string matching	translation service	taxonomic distance Wordnet
Zhishi.links	string matching		abbreviations lists
Serimi	string matching		

Table 3.4 shows that the tools require different levels of manual input. While most of the tools require manual input, two tools only are fully automated: LN2R and LD Mapper. These two tools can only be used on datasets sharing a same ontology, with LD mapper being specialized for the MusicOntology. Another tool, ObjectCoref, does not need to be configured but instead takes as input a training set that it will use to find the best appropriated configuration. The other tools require a user specification that varies in syntax, but have similar content: what kind of entities are linked, which properties are compared and which similarity computation methods are used. RKB CRS follows this scheme but includes built-in heuristics making it more efficient for linking university and conferences related datasets.

We will next discuss techniques that could be used to make tools more efficient and more automated.

Table 3.2: Systems individual matching techniques

	Individual matching		
	Data level		Knowledge Level
System	Internal	External	Internal
LN2R	maximum weighed average		sameAs InverseFunctional FunctionalProperty cardinality maxCardinality
ObjectCoref	attribute-based similarity (learned)	existing links	sameAs InverseFunctional FunctionalProperty cardinality maxCardinality
OKKAM ENS	similarity combination techniques (listed in paper)		
RKB-CRS	domain-specific metrics for citations	existing links	
LD Mapper	sum		
Silk	many		
LIMES			
KnoFuss	weighted average	sameAs transitive closure	cardinality disjointness functionality
RDF-AI	weighted average		
Zhishi.links	geographical distance		functional
			inverseFunctional
Serimi			

Table 3.3: Systems dataset matching techniques

	Dataset matching		
	Data level	Knowledge level	
System	Internal	Internal	External
LN2R	iterative sim.	??schema aware" sim. propagation	
ObjectCoref			
OKKAM ENS			
RKB-CRS			
LD Mapper			
Silk			
LIMES			
KnoFuss		Dempster-Shafer sim. propagation	third-party schema mappings
RDF-AI	sim. propagation		
Zhish.links			
Serimi	classes of interest		

Table 3.4: Systems usability

	Usability/Manual input
LN2R	Automated, works only if datasets share the same ontology
ObjectCoref	Needs a training set, then automated
OKKAM ENS	Must be run on one entity type at a time (eg. Persons, places, etc.)
RKB-CRS	A Java class has to be implemented for each pair of datasets
LD Mapper	Automated, works only for for datasets described using the MusicOntology
Silk	Link specification language: used techniques, implicit schema alignment, aggregations need to be described
LIMES	Link specification language, Web service and Web GUI
KnoFuss	Dataset description ontology needs to be written ontology alignments can be reused
RDF-AI	Dataset, techniques, pre-, and post-processing config. files need to be written
Zhishi.links	Information not available
Serimi	Automated, command line parameters.

4. Open problems and directions

As discussed in Section 1, in the field of data linking a clear and widely accepted definition of the meaning of mappings is still missing. On one side we have a wide spectrum of different techniques, capable of discovering different kinds of possible links between object descriptions, ranging from weak correspondences to strong relations of identity. On the other side, we still use (or abuse) the *owl:sameAs* relation for representing all these different links, in spite of the original meaning of this specific OWL relation. Concerning this point, a lot of work is possible towards a finer definition of the relation of identity and a richer vocabulary for representing it. Both UMBEL and SKOS can be seen as useful starting points for the definition of a shared vocabulary concerning identity and mappings. However, covering the range of possible meanings for a mapping is just part of the problem: in fact, we also need to clarify and study the correspondence between the different techniques available for instance matching and the possible meanings of their resulting mappings. To this end, some preliminary work has been proposed in **(author?)** [36], where the authors discuss a solution that allows the extraction of isomorphic statements from data without requiring their direct assertion and propose a Identity Ontology for the representation of identity. However, an analytic inquiry about the formal properties of instance matching techniques with respect to identity is still missing.

There are several potentially promising research directions concerning the development of novel matching techniques. We can particularly point out two of them. First, the growth of the Linked Data cloud provides unique possibility to use large volumes of already existing data as information sources. Moreover, the task of data linking itself becomes transformed from the traditional scenario which focuses on finding sets of mappings between two datasets to the more open task of discovering mappings within a network of many datasets. This makes the dataset matching techniques particularly important. Two examples of tools targeting these issues are idMesh[13] and ObjectCoref[26]. Second, given the variety of used schema vocabularies, methods able to overcome semantic heterogeneity and deal with information expressed using different ontologies are especially valuable.

With regards to the tools we see that there is still room for improving their automation. Most tools are semi-automated and require an extensive amount of configuration for each data linking task, while automatic tools works only in predefined domains. We identify three types of input:

- schema alignments
- matching techniques
- keys

We propose here three possible improvements to automate existing tools:

- Schema alignment could be shared using a server. By doing so they could be reused for every data linking task involving the aligned schemas.
- The selection of matching techniques is depending on the kind of data needed to be matched as some techniques are better working than others on specific data types. Information about the techniques to use could be directly attached to the schemas and thus would not need to be specified for each data linking task.
- A good amount of input is needed to specify what properties need to be compared for identifying the identity of two instances. This set of properties ensures that there

is not two instance sharing the same combination of values for these properties, they are as a key in a relational database. Statistical analysis could be used in order to automatically mine keys in RDF datasets.

5. Conclusion

After defining the data linking problem, we have presented a comprehensive survey of techniques helping to solve it. We have classified these techniques according to the criterias of granularity, type of evidence, and source of the evidence. We have presented eleven data linking systems and classified them according to which technique they use. We observed that the studied systems leave room for improvement, particularly at the dataset level of granularity. Based on these observation, we propose in deliverable D4.2 a set of techniques for the interlinking module of the Datalift platform.

References

[1] Keith Alexander, Richard Cyganiak, Michael Hausenblas, and Jun Zhao. Describing linked datasets - on the design and usage of void, the 'vocabulary of interlinked datasets'. In *Proc. WWW Linked Data on the Web Workshop (LDOW09)*, Madrid, Spain, 2009.

[2] Samur Araujo, Arjen de Vries, and Daniel Schwabe. Serimi results for oaei 2011. In *ISWC Ontology Matching workshop*, 2011.

[3] Carlo Batini and Monica Scannapieco. *Data Quality: Concepts, Methodologies and Techniques (Data-Centric Systems and Applications)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[4] Rohan Baxter, Peter Christen, and Tim Churches. A Comparison of Fast Blocking Methods for Record Linkage. In *KDD Workshop on Data Cleaning, Record Linkage, and Object Consolidation*, pages 25–27, 2003.

[5] Roberto J. Bayardo, Yiming Ma, and Ramakrishnan Srikant. Scaling up all pairs similarity search. In *16th International Conference on World Wide Web*, pages 131–140, Banff, Alberta, Canada, 2007.

[6] Jens Bleiholder and Felix Naumann. Data fusion. *ACM Comput. Surv.*, 41:1:1–1:41, January 2009.

[7] Paolo Bouquet, Heiko Stoermer, and Barbara Bazzanella. An Entity Naming System for the Semantic Web. In *Proc. 5th European Semantic Web Conference (ESWC)*, LNCS, June 2008.

[8] S. Castano, A. Ferrara, and S. Montanelli. Matching ontologies in open networked systems: Techniques and applications. *Journal on Data Semantics (JoDS)*, V, 2005.

[9] Silvana Castano, Alfio Ferrara, Davide Lorusso, Tobias Henrik Näth, and Ralf Möller. Mapping validation by probabilistic reasoning. In *Proc. 5th European semantic web conference (ESWC)*, pages 170–184, Berlin, Heidelberg, 2008. Springer-Verlag.

[10] R. Cilibrasi and P. Vitanyi. The Google similarity distance. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):370–383, 2007.

[11] William W. Cohen, Pradeep Ravikumar, and Stephen E. Fienberg. A comparison of string metrics for matching names and records. In *Proc. KDD Workshop on Data Cleaning and Object Consolidation*, 2003.

[12] Timothee Cour, Praveen Srinivasan, and Jianbo Shi. Balanced graph matching. *Advances in Neural Information Processing Systems*, 19:313–320, 2006.

[13] Philippe Cudré-Mauroux, Parisa Haghani, Michael Jost, Karl Aberer, and Hermann de Meer. idMesh: Graph-based disambiguation of linked data. In *Proc. 18th International World Wide Web Conference (WWW)*, pages 591–600, Madrid, Spain, 2009. ACM.

[14] Xin Dong, Alon Halevy, and Jayant Madhavan. Reference reconciliation in complex information spaces. In *Proc. ACM SIGMOD international conference on Management of data*, pages 85–96, New York, NY, USA, 2005. ACM.

[15] A.K. Elmagarmid, P.G. Ipeirotis, and V.S. Verykios. Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 19(1):1–16, January 2007.

[16] Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching*. Springer-Verlag, Heidelberg (DE), 2007.

[17] Ivan P. Fellegi and Alan B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, 1969.

[18] Hugh Glaser, Ian Millard, and Afraz Jaffri. Rkbexplorer.com: A knowledge driven infrastructure for linked data providers. In *Proc. Xth European Semantic Web Conference (ESWC)*, pages 797–801, 2008.

[19] Risto Gligorov, Zharko Aleksovski, Warner ten Kate, and Frank van Harmelen. Using Google distance to weight approximate ontology matches. In *Proc. 16th International World Wide Web Conference (WWW)*, pages 767–775, Banff, Canada, 2007.

[20] Jorge Gracia, Mathieu d'Aquin, and Eduardo Mena. Large scale integration of senses for the Semantic Web. In *Proc. 18th International World Wide Web Conference (WWW)*, Madrid, Spain, 2009.

[21] Jorge Gracia and Eduardo Mena. Matching with CIDER: Evaluation report for the OAEI 2008. In *Proc. 3rd ISWC Ontology Matching Workshop*, Karlsruhe, Germany, 2008.

[22] Nicola Guarino and Chris Welty. *The Semantics of Relationships: An Interdisciplinary Perspective*, chapter Identity and Subsumption, pages 111–125. Kluwer, Dordrecht, 2002.

[23] Harry Halpin, Patrick J. Hayes, James P. McCusker, Deborah L. McGuinness, and Henry S. Thompson. When owl:sameas isn't the same: An analysis of identity in linked data. In *Proc. 9th International Semantic Web Conference (ISWC)*, pages 305–320, Shanghai, China, 2010.

[24] Oktie Hassanzadeh, Lipyeow Lim, Anastasios Kementsietsidis, and Min Wang. A declarative framework for semantic link discovery over relational data. In *Proc. 18th international World wide web conference*, pages 1101–1102, New York, NY, USA, 2009. ACM.

[25] W. Hu, J. Chen, and Y. Qu. A self-training approach for resolving object coreference on the semantic web. In *Proc. WWW Conference, Hyderabad (IN)*. ACM, 2011.

[26] Wei Hu, Jianfeng Chen, and Yuzhong Qu. A self-training approach for resolving object coreference on the semantic web. In *Proc. 20th International World Wide Web Conference (WWW)*, 2011.

[27] Ekaterini Ioannou, Claudia Niederée, and Wolfgang Nejdl. Probabilistic entity linkage for heterogeneous information spaces. In *Proc. 20th International Conference on Advanced Information Systems Engineering (CAiSE)*, pages 556–570, 2008.

[28] Ekaterini Ioannou, Claudia Niederée, Yannis Velegrakis, Nicolas Bonvin, Adrian Mocan, Georges Papadakis, Elena Demidova, Julien Gaugaz, and Nataliya Rassadko. D3.1 Intelligent entity matching and ranking. Technical report, OKKAM, 2010.

[29] Afraz Jaffri, Hugh Glaser, and Ian Millard. URI disambiguation in the context of Linked Data. In *Proc. WWW Linking Data On the Web workshop*, 2008.

[30] Yves R. Jean-Mary, E. Patrick Shironoshita, and Mansur R. Kabuka. Ontology matching with semantic verification. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7, 2009.

[31] H. Köpcke, A. Thor, and E. Rahm. Evaluation of entity resolution approaches on real-world match problems. In *Proc. 36th Intl. Conference on Very Large Databases (VLDB)*, 2010.

[32] Hanna Köpcke and Erhard Rahm. Frameworks for entity matching: A comparison. *Data Knowl. Eng.*, 69:197–210, February 2010.

[33] N. Koudas, S. Sarawagi, and D. Srivastava. Record linkage: similarity measures and algorithms. In *Proc. ACM SIGMOD international conference on Management of data*, pages 802–803. ACM, 2006.

[34] Juanzi Li, Jie Tang, Yi Li, and Qiong Luo. RiMOM: A dynamic multistrategy ontology alignment framework. *IEEE Transactions on Knowledge and Data Engineering*, 21(8):1218–1232, 2009.

[35] Vanessa Lopez, Andriy Nikolov, Miriam Fernandez, Marta Sabou, Victoria Uren, and Enrico Motta. Merging and ranking answers in the Semantic Web: The wisdom of crowds. In *Proc. 4th Asian Semantic Web Conference (ASWC)*, pages 135–152, 2009.

[36] J. McCusker and D. McGuinness. Towards identity in linked data. In *Proc. 7th OWL Experiences and Directions Annual Workshop*, 2010.

[37] Christian Meilicke, Johanna Völker, and Heiner Stuckenschmidt. Learning disjointness for debugging mappings between lightweight ontologies. In *Proc. 16th international conference on Knowledge Engineering (EKAW)*, EKAW '08, pages 93–108, Berlin, Heidelberg, 2008. Springer-Verlag.

[38] Sergey Melnik. Similarity flooding: a versatile graph matching algorithm. In *Proc. 18th International Conference on Data Engineering (ICDE), San Jose (CA US)*, 2002.

[39] George Miller. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

[40] Axel-Cyrille Ngonga Ngomo and Sören Auer. LIMES - a time-efficient approach for large-scale link discovery on the web of data. In *Proc. International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

[41] Axel-Cyrille Ngonga Ngomo. A Time-Efficient Hybrid Approach to Link Discovery. In *6th International Workshop on Ontology Matching*, Bonn, Germany, 2011.

[42] Andriy Nikolov, Victoria Uren, and Enrico Motta. Data linking: Capturing and utilising implicit schema-level relations. In *Proc. 3rd Workshop on Linked Data on the Web (LDOW)*, Raleigh, USA, 2010.

[43] Andriy Nikolov, Victoria Uren, Enrico Motta, and Anne de Roeck. Integration of semantically annotated data by the KnoFuss architecture. In *Proc. 16th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*, pages 265–274, Acitrezza, Italy, 2008.

[44] Andriy Nikolov, Victoria Uren, Enrico Motta, and Anne de Roeck. Refining instance coreferencing results using belief propagation. In *Proc. 3rd Asian Semantic Web Conference (ASWC)*, pages 405–419, Bangkok, Thailand, 2008.

[45] Xing Niu, Shu Rong, Yunlong Zhang, and Haofen Wang. Zhishi.links results for oaei 2011. In *Proc. ISWC Ontology Matching workshop*, 2011.

[46] Jan Noessner, Mathias Niepert, Christian Meilicke, and Heiner Stuckenschmidt. Leveraging terminological structure for object reconciliation. In *Proc. 7th Extended Semantic Web Conference (ESWC)*, pages 334–348, Heraklion, Crete, Greece, 2010.

[47] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, San Francisco, 1988.

[48] Erhard Rahm and Philip A. Bernstein. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10(4):334–350, 2001.

[49] Yves Raimond, Christopher Sutton, and Mark Sandler. Automatic interlinking of music datasets on the semantic web. In *Proc. WWW Linking Data On the Web workshop*, 2008.

[50] Fatiha Saïs, Nathalie Pernelle, and Marie-Christine Rousset. L2r: A logical method for reference reconciliation. In *Proc. AAAI*, pages 329–334, 2007.

[51] Fatiha Saïs, Nathalie Pernelle, and Marie-Christine Rousset. Combining a logical and a numerical method for data reconciliation. *Journal of Data Semantics*, 12, 2008.

[52] François Scharffe, Yanbin Liu, and Chunguang Zhou. RDF-AI: an architecture for RDF datasets matching, fusion and interlink. In *Proc. IJCAI workshop on Identity and Reference in Knowledge Representation*, 2009.

[53] Alexander Schrijver. *Theory of Linear and Integer Programming*. Wiley, 1998.

[54] Glenn Shafer. *A mathematical theory of evidence*. Princeton University Press, 1976.

[55] Prakash P. Shenoy. *Fuzzy logic for the management of uncertainty*, chapter Valuation-based systems: a framework for managing uncertainty in expert systems, pages 83–104. John Wiley & Sons, Inc., New York, NY, USA, 1992.

[56] D. Song and J. Heflin. Automatically generating data linkages using a domain-independent candidate selection approach. In *Proc. 10th International Semantic Web Conference (ISWC), Bonn (DE)*, 2011.

[57] Giorgos Stoilos, Giorgos Stamou, and Stefanos Kollias. A string metric for ontology alignment. In *Proc. 4th International Semantic Web Conference (ISWC)*, pages 624–637, Galway, Ireland, 2005.

[58] Giovanni Tummarello, Renaud Delbru, and Eyal Oren. Sindice.com: Weaving the open linked data. In *Proc. 6th International Semantic Web Conference (ISWC/ASWC)*, pages 552–565, Busan, Korea, 2007.

[59] Octavian Udrea, Lise Getoor, and Renée J. Miller. Leveraging data and structure in ontology integration. In *Proc. SIGMOD*, pages 449–460, Beijing, China, 2007.

[60] Julius Volz, Christian Bizer, and Martin Gaedke. Web of data link maintenance protocol. Protocol specification, Frei Universität Berlin, 2009.

[61] Julius Volz, Christian Bizer, Martin Gaedke, and Georgi Kobilarov. Discovering and maintaining links on the web of data. In *Proc. 8th International Semantic Web Conference (ISWC)*, pages 650–665, Washington, DC, USA, 2009.

[62] W.E Winkler. Overview of record linkage and current research directions. Technical Report 2006-2, Statistical Research Division. U.S. Census Bureau, 2006.

[63] Chuan Xiao, Wei Wang, Xuemin Lin, and Jeffrey Xu Yu. Efficient similarity joins for near duplicate detection. In *17th International Conference on World Wide Web*, pages 131–140, New York, NY, USA, 2008.