ANR-17-CE23-0007

ELKER

Enhancing link keys: extraction and reasoning
Étendre les clés de liage: extraction et raisonnement

D1.3 Quality measures and link key extraction strategies

Coordinator: Jérôme Euzenat
With contributions from: Manuel Atencia, Jérôme David, Jérôme Euzenat

Quality reviewer:	Specify quality controller
Reference:	ELKER/D1.3/v1.0
Project:	ELKER ANR-17-CE23-0007
Date:	December 8, 2021
Version:	1.0
State:	final
Destination:	public

Executive Summary

With the distributed publication of large amounts of RDF data on the web, it is necessary to find entities occurring in several data sets and to express their identity. This is expressed through owl:sameAs assertions also called links. In the perspective of automating link generation, efforts are concentrated on extracting *link specifications*.

There exist two main types of link specifications: logical and numerical link specifications. Numerical link specifications roughly select pairs of entities to link based on a similarity measure computed between them. Link specifications express conditions under which instances of two classes of different RDF data sets may be linked. There exist algorithms for extracting link specification candidates from RDF data and measures have been defined to evaluate their quality.

However, very often, it may be necessary to use several link specifications, even on the same pair of classes, for retrieving a more complete link set.

Thus, we aim at combining different link specifications. For that purpose, we introduce operators to combine link specifications, investigate their relations and extend measures to evaluate their quality. Operators are classical boolean operators (conjunction, disjunction, negation) and operators that complement an existing specification with another one avoiding to link the same entity several times. The semantics of these operators is given by the link sets that they generate allowing to provide a uniform semantics over all specifications.

The relationship between these operators is formally assessed. We then go on further investigating this approach more specifically as applied to link keys. More specific relations with respect to the algebraic link key constructors are also discussed.

We discuss strategies to extract the best uniform compound link key candidates, i.e. compound candidates using a single operator. In particular, we extend link key candidate extraction to deal with disjunctions of link key candidates. We show that all candidates belong to the set of antichains of the link key candidates lattice and that they are themselves organised in a lattice. We define heuristics in order to identify the best candidates with respect to the available quality measures.

Finally, we report on experiments with such operators on standard data sets. We show that the extracted disjunctions improve the F-measure of individual link key candidates. However, the experiments demonstrate that further improvements are needed, either by combining both heuristics, or by developing more suitable unsupervised measures.

It would be interesting to specify the compound link key extraction problem directly within formal concept analysis and compare the equivalence and benefits of the proposed approach.

This deliverable generalises the results published in [Atencia et al. 2019].

Document Information

Project number	ANR-17-CE23-0007	**Acronym**	ELKER
Full Title	Enhancing link keys: extraction and reasoning Étendre les clés de liage: extraction et raisonnement		
Project URL	`https://project.inria.fr/elker/`		
Document URL			

Deliverable	**Number**	1.3	**Title**	Quality measures and link key extraction strategies
Work Package	**Number**	1	**Title**	Link key extraction

Date of Delivery	**Contractual**	M30	**Actual**	26/07/2021
Status	final		final ☒	
Nature	prototype ☐ report ☒ dissemination ☐			
Dissemination level	public ☒ consortium ☐			

Authors (Partner)	Manuel Atencia, Jérôme David, Jérôme Euzenat			
Resp. Author	**Name**	Jérôme Euzenat	**E-mail**	Jerome.Euzenat@inria.fr
	Partner	LIG		

Abstract (for dissemination)	This report address the interaction between quality measures and link key extraction strategies. More generally, it considers the possibility to combine any kind of link specifications and to extract compound link keys which maximise quality.
Keywords	link specifications, link keys, composition, compound link key, compound link key extraction, quality measures, coverage, discriminability, functionality

Version Log			
Issue Date	**Rev No.**	**Author**	**Change**
03/05/2019	1	J. Euzenat	Initial version
20/06/2019	2	J. Euzenat	Filed with additional comments
19/10/2020	3	J. Euzenat	Integrated K-Cap paper writing
22/10/2020	4	J. Euzenat	Reorganised and generalised to link specifications
19/11/2020	5	J. Euzenat	Revised the whole document
31/03/2021	6	J. David	Revised and commented
02/04/2021	7	J. Euzenat	Addressed comments
26/07/2021	8	J. David	Checked all remaining issues

Table of Contents

1. Introduction

Finding links across linked open data sets is an important task as it enables data interoperability. Different approaches to data interlinking have been proposed [Ferrara et al. 2011; Nentwig et al. 2017]. One of the these is based on link keys [Atencia et al. 2014]. Link keys generalise relational keys to the case of two RDF data sets. They express conditions under which two instances of different data sets may be linked. Algorithms have been proposed for extracting link key candidates from RDF data and, for selecting the best candidates, supervised and unsupervised measures have been developed [Atencia et al. 2014; Atencia et al. 2020].

However, one single link key, even the best one, may not be sufficient to generate all links in some data sets. This is simply the case of data sources covering different concepts. This may also be useful if the data related to a particular class is the result of aggregating different sources that use different properties: there may be several different ways to generate links. Thus, instead of selecting one single best link key candidate, it could be worth selecting the best combination of link key candidates[1].

This report addresses the specific problem of defining and extracting compound link specification candidates from two RDF data sets. For that purpose, we will:

- Define precisely link specifications;
- Identify and define more precisely what is meant by combining link specifications;
- Provide the semantics of compound link specifications, i.e. specify which links they generate;
- Study the relations between various compound link specifications and more precisely compound link key expression;
- Extend available evaluation measures to compound link specifications;
- Discuss strategies for extracting the best disjunction of link key candidates;
- Evaluate these strategies experimentally.

Hence this report contributes on both theoretical and practical aspects. On the theoretical aspect we provide definitions for link specifications and compound link specifications and investigate their relations with numeric link specifications and link keys. On the practical side, we show that link key extraction methods can be extended to disjunction of link keys, and that practical techniques can be used for that. This is challenging because of the large number of non redundant disjunctions of link key candidates (potentially 2^n where n is the number of link key candidates).

After considering related work in data interlinking, and especially approaches that consider combining link specifications (Chapter 2), we more thoroughly generalise link specifications and introduce link keys (Chapter 3) and measures to evaluate them (Chapter 4). Then, we introduce composition operators (Chapter 5) and precise their applications to link keys (Chapter 6). We discuss strategies for extracting uniform compound link keys (Chapter 7) and evaluate them on several data sets (Chapter 8).

[1] We use *combination* as a general informal term, more precisely, we define *composition operators* and how they build *compound link specifications*.

2. Related work

Data interlinking or *entity matching* refers to the process of finding pairs of IRIs representing the same resource in different RDF data sets [Ferrara et al. 2011; Christen 2012; Nentwig et al. 2017]. The result of this process is a set of links, which may be added to the data sets by relating the corresponding IRIs with the owl:sameAs property. Data interlinking can be defined as follows: given two sets of individual identifiers I_D and $I_{D'}$ from two data sets D and D', find the set L of pairs of identifiers $\langle o, o' \rangle \in I_D \times I_{D'}$ such that o and o' denote the same resource.

Links are usually produced by using a framework, such as SILK [Volz et al. 2009] and LIMES [Ngonga Ngomo and Auer 2011], processing *link specifications*. Link specifications indicate the conditions for two IRIs to be linked. They may be directly defined by users or (semi-)automatically extracted. The present work is concerned with the combination and evaluation of several link specifications together.

Most methods roughly compute a *numerical specification* $\langle \sigma, \theta \rangle$ made of a similarity measure σ between the entities to be linked and a threshold θ. It is assumed that, if two entities are very similar, then they are likely the same. Hence, such specifications may generate links through (adapted from [Sherif et al. 2017]):

$$L_{\sigma,\theta}^{D,D'} = \{\langle o, o' \rangle \in I_D \times I_{D'}; \sigma(o, o') \geq \theta\}$$

Wombat [Sherif et al. 2017] provides a way of exploring the space of such link specifications, starting with atomic similarity between pairs of datatype property values. It is able to learn, in a supervised manner, conjunction, disjunction and difference of link specifications with the semantics that we will consider in this paper. SILK, via the ActiveGenLink algorithm [Isele and Bizer 2013], composes similarity components (similarity metrics), but not full link specifications as it is done in Wombat. Unlike Wombat and SILK, the approach presented here is fully unsupervised: it does not need any sample links, neither property not class alignment.

Logical link specifications are logical axioms of which the links are consequences. Unlike numerical specifications, they can be combined with other kinds of knowledge, such as ontologies and ontology alignments, to infer links by using reasoning [Saïs et al. 2007; Al-Bakri et al. 2015; Al-Bakri et al. 2016; Hogan et al. 2012]. Logical link specifications do not incorporate similarity metrics natively but such metrics, if necessary, may be handled separately through specific rules or in a data preprocessing step. There are various ways to approach the definition of such specifications [Saïs et al. 2007; Al-Bakri et al. 2015; Al-Bakri et al. 2016; Hogan et al. 2012].

Key-based specifications fall into the category of logical link specifications. Key-based approaches typically extract keys from RDF data sets and combine them with ontology alignments for interlinking [Symeonidou et al. 2014; Achichi et al. 2016; Farah et al. 2017; Atencia et al. 2012].

Here we deal with the combination of *link keys*, a specific type of logical link specification. *Link keys* can be seen as the generalisation of relational database keys to the case of two different data sets and to the specifics of RDF [Euzenat and Shvaiko 2013; Atencia et al. 2014]. An example of a link key is:

$$\{\langle \mathsf{auteur}, \mathsf{creator} \rangle\}\{\langle \mathsf{titre}, \mathsf{title} \rangle\} \; \mathit{linkkey} \; \langle \mathsf{Livre}, \mathsf{Book} \rangle$$

stating that whenever an instance of the class Livre has the same values for the property auteur as an instance of the class Book has for the property creator and they share at least one

value for their properties titre and title, then they denote the same entity. A link key may be thought of as a pair of aligned keys, but the relation between link keys and keys is more subtle, as we explain below.

The key-based approaches [Achichi et al. 2016; Farah et al. 2017] aim at using a key extraction algorithm [Symeonidou et al. 2014] to extract pairs of keys that can be used as link specifications. They extract IN-keys, i.e. keys based on sharing one value between properties, which hold in both source and target data sets. It is assumed that both data sets are described using the same ontology or, more precisely, the system only looks for keys based on the vocabulary common to the two data sets. In this case, the extracted IN-keys, though not equal, mostly correspond to strong IN-link keys (i.e. link keys that are made up of keys) [Atencia et al. 2021], and not to weak IN-link keys, which are more general and are the kind of link keys extracted in [Atencia et al. 2014].

KeyRanker [Farah et al. 2017] describes a method for selecting and assembling a disjunction of several such pairs of keys. The selection always starts with the most covering candidate but further key pairs are selected based on the marginal covering contribution of the added candidate.

There is no necessary correspondence between keys and link keys: there may exist keys which are part of no link key and link keys which do not rely on keys [Euzenat and Shvaiko 2013, Example 5.38, p116]. Hence, searching for keys to be eventually turned into link keys may fail [Atencia et al. 2021].

A technique for directly extracting link keys between two classes from two RDF data sets has been proposed [Atencia et al. 2014]. Unlike key-based approaches, the proposed algorithm does not require an initial alignment between properties of both data sets nor makes the assumption of common vocabularies. It also avoids the generation of keys that are specific to one data set only. It first extracts link key candidates from the data and then uses measures of the quality of these candidates in order to select the one to apply. Finally, although it can take advantage of them, it does not rely on supervised machine learning, and, hence, does not need training links.

This deliverable aims at exploring the potential for combining link specifications through defining their syntax and semantics. It more specifically attempts at extending the link key candidate extraction approach to the combination of extracted link keys and at providing insights about how this can be achieved.

3. Preliminaries

3.1 RDF Data sets

First, we provide a definition of an RDF data set sufficient for our purposes.

Definition 1 (RDF data set)**.** *Let U be a set of IRIs, B a set of blank nodes and L a set of literals. An* RDF data set *is a set of triples from $(U \cup B) \times U \times (U \cup B \cup L)$.*

Given an RDF data set D, we denote by U_D, B_D and L_D, respectively, the sets of IRIs, blank nodes and literals present in D.

RDF triple components are identified by their respective positions $\langle subject, property, object \rangle$. A special property, very often used in RDF data sets, is rdf:type, declaring that an individual belongs to a particular class, e.g. $\langle$o1:z1, rdf:type, o1:Person$\rangle$. The following definition makes explicit the distinction between individuals, properties and classes of an RDF data set.

Definition 2 (Identifiers of an RDF data set)**.** *Let D be an RDF data set, the sets I_D of* individual identifiers*, P_D of* datatype property identifiers*, R_D of* object property identifiers *and C_D of* class identifiers *in D, are defined as follows:*

$$
\begin{aligned}
I_D &= \{o \in U_D | \exists \langle o, p, u \rangle \in D \text{ or } \exists \langle u, p, o \rangle \in D\} \\
P_D &= \{p \in U_D | \exists u \in L_D; \langle o, p, u \rangle \in D\} \\
R_D &= \{r \in U_D | \exists u \in U_D \cup B_D; \langle o, r, u \rangle \in D\} \\
C_D &= \{c \in U_D | \exists \langle o, \textsf{rdf:type}, c \rangle \in D\}
\end{aligned}
$$

The vocabulary of "class", "datatype property", "object property" and "individual" is used according to their meaning in RDFS [Brickley and Guha 2014] and OWL [Motik et al. 2012]. From these we define the signature of a data set.

Definition 3 (Signature of an RDF data set)**.** *The* signature *of an RDF data set D is the tuple $\langle R_D, P_D, C_D \rangle$.*

We denote by $c^D = \{t \in I_D | \langle t, \textsf{rdf:type}, c \rangle \in D\}$ the set of instances of $c \in C_D$ in data set D. In RDF, an individual may have several different values for the same property. For a datatype property $p \in P_D$, we denote by $p^D(o) = \{v \in L_D | \langle o, p, v \rangle \in D\}$ the set of values of property p for object o in data set D. Similarly, for the object property $r \in R_D$, we have $r^D(o) = \{u \in I_D | \langle o, r, u \rangle \in D\}$.

Example 1 (RDF Data set)**.** *Let us consider the two very simple data sets of Table 3.1.*[1] *The table displays the instances of two classes of two data sets D and D'. The signature of D is $\langle \{\}, \{$prénom, datenaiss, post, bât.$\}, \{$Employé$\} \rangle$; that of D' is $\langle \{\}, \{$firstname, birthdate, position, building$\}, \{$Staff$\} \rangle$. This type of example may occur when the second data set (D') is the result of the merge of two heterogeneous data sources (one using* birthdate *property and the other using* building *property).*

[1] For the sake of readability, we represent these RDF data sets as simple relational tables without multiple values or object references. They are only here to explain the problems and this makes it easier to compare them.

D (Employé)				
id	prenom	datenaiss	poste	bât.
i_2	Paul	1967	Dir.	B2
i_3	Mary	1963	Dir.	B1
i_4	John	1963	Pr.	B1
i_6	Bill	1980	Pr.	B1
i_7	Ana	1947	Dir.	B2
i_8	John	1967	Pr.	B2

D' (Staff)				
firstname	birthdate	position	building	id
Paul		Dir.	B2	z_2
Mary		Dir.	B1	z_3
John		Pr.	B1	z_4
William	1980	Pr.		z_6
Ana	1947	Dir.		z_7
John	1967	Pr.		z_8

Table 3.1: Instances of the classes Employé and Staff of two RDF data sets D and D'.

3.2 Link specifications

In order to generate links between two data sets, we introduce the notion of link specification.

Definition 4 (Link specification)**.** *A link specification is a function which associates to two data sets D and D' a set of links $L_s(D, D') \subseteq I_D \times I_{D'}$.*

Hence, for a given pair of data sets, each link specification, whatever its type, has an image within the sets of link sets. Below we note $L_s^{D,D'}$ for $L_s(D, D')$ the set of links generated by the specification L. Moreover, these specifications can be organised in a partial order anti-monotonic to link set inclusion called specification subsumption.

Definition 5 (Extensional specification subsumption)**.** *Given two link specifications L_1 and L_2, L_1 is (extensionally) subsumed by L_2 (noted $L_1 \preceq L_2$) if*

$$\forall D, D', \quad L_1^{D,D'} \supseteq L_2^{D,D'}.$$

In [Atencia et al. 2019], we used the notion of extended subsumption, instead of intentional subsumption:

Definition 6 (Extended subsumption of link specifications)**.** *Let D and D' be two data sets. Let L_1 and L_2 be two link specifications over their signatures. We say that L_2* (extensively) subsumes *L_1, written $L_1 \preceq^{D,D'} L_2$, if $L_1^{D,D'} \supseteq L_2^{D,D'}$.*

The relation between these two notions is thus the following:

Property 1.

$$\forall D, D', L_1 \preceq^{D,D'} L_2 \quad \equiv \quad L_1 \preceq L_2$$

In the following, the "D, D'" exponent may be avoided since we only compare specifications on the same data sets. We write $k \prec h$, when $k \preceq h$ but not $h \preceq k$, $k \simeq h$, when $k \preceq h$ and $h \preceq k$, and $k \dot{\prec} h$, when $k \prec h$ and there exist no l such that $k \prec l \prec h$.

Until now, we opposed two types of "link specifications": numerical and logical specifications. A numerical specification $\langle \sigma, \theta \rangle$ is made of a similarity measure σ between the entities to be linked and a threshold θ. They assume that, if two entities are very similar, they are likely the same.

This can be expressed by:

$$L_{\sigma,\theta}^{D,D'} = \{\langle o, o' \rangle \in I_D \times I_{D'}; \sigma(o, o') \geq \theta\}$$

It is possible to define a syntactic subsumption between numerical link specifications, for instance by:

$$\langle \sigma, \theta \rangle \leq \langle \sigma, \theta' \rangle \text{ if } \theta \geq \theta'$$

Given two link specifications s and s', it can be checked that:

$$\text{if } s \leq s', \text{then } \forall D, D', L_s^{D,D'} \supseteq L_{s'}^{D,D'}$$

and thus:

$$s \preceq s'$$

This is where formal concept analysis comes handy: from this, it is possible to define closure operations and concepts which will form a sub-lattice of $\langle 2^{I_D \times I_{D'}}, \subseteq \rangle$. This is especially interesting for numeric specifications since, from a potentially infinite set of specifications due to the dense nature of θ, it is possible to isolate a finite set of concepts[2].

3.3 Link keys

Link keys specify the pairs of properties to compare for deciding whether individuals of two classes of two different data sets have to be linked. We first give the definition of a link key expression.

Definition 7 (Link key expression [Atencia et al. 2020])**.** *A* link key expression *over two signatures $\langle R, P, C \rangle$ and $\langle R', P', C' \rangle$ is an element of $2^{(P \times P') \cup (R \times R')} \times 2^{(P \times P') \cup (R \times R')} \times (C \times C')$, i.e.*

$$\langle \{\langle p_i, p_i' \rangle\}_{i \in EQ}, \{\langle q_j, q_j' \rangle\}_{j \in IN}, \langle c, c' \rangle \rangle$$

such that EQ and IN are (possibly empty) finite sets of indices.

Example 2 (Link key expressions)**.** *Given the signatures of the data sets D and D' of Example 1, the following are examples of link key expressions:*

$$k = \langle \{\langle \mathit{datenaiss}, \mathit{birthdate} \rangle\}, \{\}, \langle \mathit{Employé}, \mathit{Staff} \rangle \rangle$$
$$h = \langle \{\langle \mathit{datenaiss}, \mathit{birthdate} \rangle\}, \{\langle \mathit{poste}, \mathit{position} \rangle\}, \langle \mathit{Employé}, \mathit{Staff} \rangle \rangle$$
$$l = \langle \{\langle \mathit{datenaiss}, \mathit{birthdate} \rangle, \langle \mathit{poste}, \mathit{position} \rangle\}, \{\langle \mathit{poste}, \mathit{position} \rangle\}, \langle \mathit{Employé}, \mathit{Staff} \rangle \rangle$$

Link key expressions may be used to generate links between RDF data sets.

Definition 8 (Link set generated by a link key expression [Atencia et al. 2020])**.** *Let D and D' be two data sets and $k = \langle \{\langle p_i, p_i' \rangle\}_{i \in EQ}, \{\langle q_j, q_j' \rangle\}_{j \in IN}, \langle c, c' \rangle \rangle$ be a link key expression over their signatures. The* link set generated by *k for D and D' is the subset $L_k^{D,D'} \subseteq c^D \times c'^{D'}$ defined as:*

$$\langle o, o' \rangle \in L_k^{D,D'} \text{ iff } \begin{cases} p_i^D(o) = p_i'^{D'}(o') & \text{for all } i \in EQ, \text{ and} \\ q_j^D(o) \cap q_j'^{D'}(o') \neq \emptyset & \text{for all } j \in IN \end{cases}$$

[2]There is however a subtlety, in that the representation of the intent of the concept should be the greatest of those generating the same set of links. This is in fact, the highest threshold not accepting links of a certain threshold θ. Unfortunately, due to the at least dense nature of the thetas, there is no 'supremum', but we can take this θ as an upper bound and note the greatest numerical specification as θ-something.

Example 3 (Generated link sets)**.** *Given the link key expressions of Example 2 and the data sets D and D' of Example 1, the generated links are:*

$$L_k^{D,D'} = \{\langle i_6, z_6\rangle, \langle i_7, z_7\rangle, \langle i_8, z_8\rangle, \langle i_2, z_8\rangle\}$$
$$L_l^{D,D'} = L_h^{D,D'} = \{\langle i_6, z_6\rangle, \langle i_7, z_7\rangle, \langle i_8, z_8\rangle\}$$

The meaning of k is the set of all pairs of Employé/Staff members having exactly the same datenais/birthdate. That of l and h, are those which, in addition, share at least one poste/-position. They generate the same links for D and D' because the data set is simple. If Ana *(z_7) also held the position of* Dean*, then $\langle i_7, z_7\rangle$ would be in $L_h^{D,D'}$, but not in $L_l^{D,D'}$. This shows that $L_l \preceq^{D,D'} L_h$ but $L_l \not\preceq L_h$.*

According to the Definition 8, statements like i_2 owl:sameAs *i_8 are no considered as generated links because they belong to the same data set. The links, like this one and many others, entailed by the generation of the* owl:sameAs *links are left to the application interpretation.*

Link key expressions may be combined through subsumption, meet or join.

Definition 9 (Subsumption, meet and join of link key expressions [Atencia et al. 2020])**.** *Let $k = \langle E, I, \langle c, c'\rangle\rangle$ and $h = \langle F, J, \langle c, c'\rangle\rangle$ be two link key expressions over the same pair of signatures $\langle R, P, C\rangle$ and $\langle R', P', C'\rangle$. k is said* (intentionally) subsumed by *h, and written $k \trianglelefteq h$, if $E \subseteq F$ and $I \subseteq J$. Additionally, the* meet *and* join *of k and h, denoted by $k \vartriangle h$ and $k \triangledown h$, respectively, are defined as follows:*

$$k \vartriangle h = \langle E \cap F, I \cap J, \langle c, c'\rangle\rangle$$
$$k \triangledown h = \langle E \cup F, I \cup J, \langle c, c'\rangle\rangle$$

Subsumption of link key expressions is contravariant with the inclusion of their generated link sets. This reflects the fact that the more constraints there are, the less links satisfy them.

The (extensive) subsumption is thus an extension of subsumption (Property 2):

Property 2. *If $k \trianglelefteq h$, then $k \preceq h$*

Proof. This is [Atencia et al. 2020, Lemma 1(1)]. □

There exists a procedure to extract a small subset of link key expressions, called *link key candidates* [Atencia et al. 2014]. This procedure has been further described in terms of formal concept analysis [Ganter and Wille 1999] and it has been shown that the set of link key candidates form a (concept) lattice [Atencia et al. 2020; Euzenat et al. 2019]. This lattice for the data set of Example 1 is depicted in Figure 3.1. These are the candidates that we want to combine and evaluate.

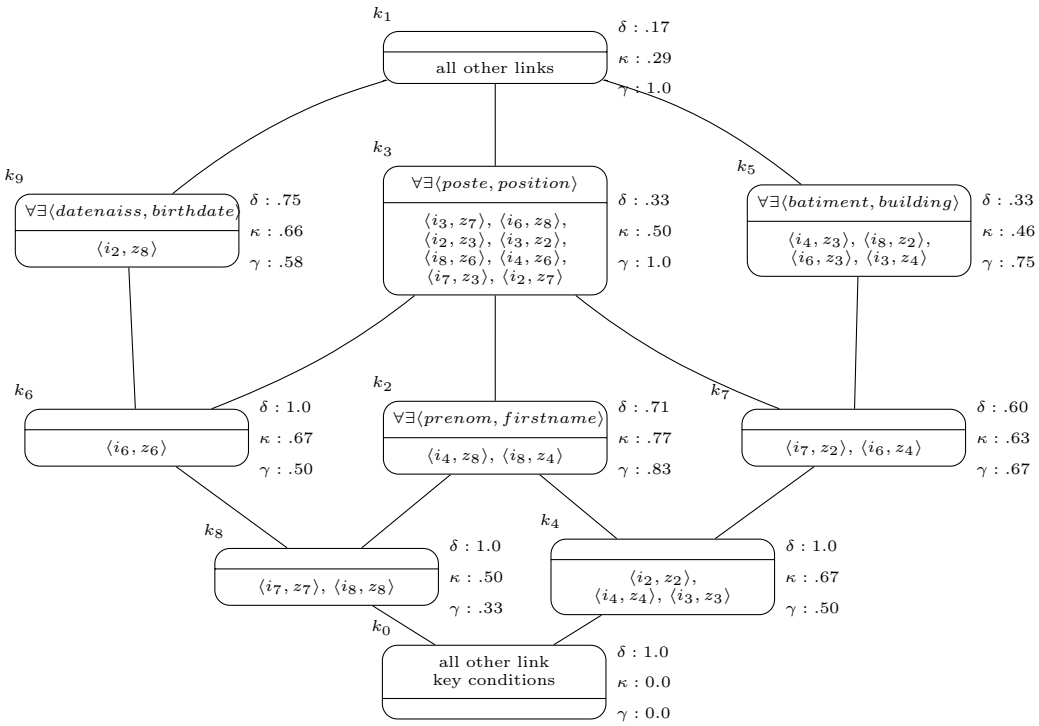

Figure 3.1: Lattice $\langle K, \trianglelefteq \rangle$ of extracted link key candidates from the data sets D and D' of Example 1 (δ=discriminability, γ=coverage, κ=harmonic mean between them). The notation $\forall\exists$ means that the pair of properties is both in the conditions indexed by IN and EQ. The lattice is drawn with the conventions of formal concept analysis [Ganter and Wille 1999].

4. Evaluation measures

When comparing link specifications in a non supervised way, i.e. without any set of correct links to compare to, evaluation measures are defined.

4.1 Discriminability and coverage

In order to select the best link key candidate, discriminability and coverage measures have been proposed [Atencia et al. 2014]. These are unsupervised measures, i.e. they do not require any link as input. We reformulate them directly with respect to the generated link sets as it will be useful later.

If L_s is a set of links between D and D', then we define

$$\pi(L_s) = \{o \in D; \langle o, o' \rangle \in L_s\} \text{ and } \pi'(L_s) = \{o' \in D'; \langle o, o' \rangle \in L_s\}$$

and a generalisation of projection:

$$L_{s,R\ n}(D) = \{a; |\{b; \langle a, b \rangle \in L_s\}|\ R\ n\} \quad L_{s,R\ n}(D') = \{b; |\{a; \langle a, b \rangle \in L_s\}|\ R\ n\}$$

with $R \in \{\leq, =, \geq\}$ and $n \in \mathbb{N}^+$. Projection corresponds to the operations with ≥ 1, functionality with ≤ 1. Discriminability measures how close the links generated by a link specification are from a one-to-one mapping. The idea behind discriminability is to disqualify link specifications that would link the same entity of one data set to several entities from the other data set (e.g. $\langle\{\langle \mathsf{firstName}, \mathsf{givenName}\rangle\}, \{\}, \langle \mathsf{Person}, \mathsf{Person}\rangle\rangle$). Such link keys show a low discriminability on data set for which they generate such links.

Definition 10 (Discriminability)**.** *Let s be a link specification,*

$$\delta^{D,D'}(s) = \begin{cases} 1.0 & \text{if } L_s^{D,D'} = \varnothing \\ \dfrac{min(|\pi(L_s^{D,D'})|, |\pi'(L_s^{D,D'})|)}{|L_s^{D,D'}|} & \text{otherwise} \end{cases}$$

or

$$\delta^{D,D'}(s) = \frac{min(|L_{s,\geq 1}^{D,D'}(D)|, |L_{s,\geq 1}^{D,D'}(D')|)}{|L_s^{D,D'}|}$$

Coverage measures how complete a link specification is with respect to the data sets, i.e. the proportion of instances of both classes that would be linked by the link specification:

Definition 11 (Coverage)**.** *Let s be a link specification between two classes c and c'*

$$\gamma^{D,D'}(s) = \begin{cases} 1.0 & \text{if } c^D = c'^{D'} = \varnothing \\ \dfrac{|\pi(L_s^{D,D'}) \cup \pi'(L_s^{D,D'})|}{|c^D \cup c'^{D'}|} & \text{otherwise} \end{cases}$$

or (if c and c' are disjoint)

$$\gamma^{D,D'}(s) = \frac{|L_{s,\geq 1}^{D,D'}(D)| + |L_{s,\geq 1}^{D,D'}(D')|}{|c^D| + |c'^{D'}|}$$

The coverage measure always favours the most general link specifications:

Property 3. *If* $s_1 \preceq^{D,D'} s_2$*, then* $\gamma^{D,D'}(s_1) \geq \gamma^{D,D'}(s_2)$

Proof. If $s_1 \preceq^{D,D'} s_2$, then $L_{s_1}^{D,D'} \supseteq L_{s_2}^{D,D'}$. For coverage, $\{a|\langle a,b\rangle \in L_{s_1}^{D,D'}\} \cup \{b|\langle a,b\rangle \in L_{s_1}^{D,D'}\} \supseteq \{a|\langle a,b\rangle \in L_{s_2}^{D,D'}\} \cup \{b|\langle a,b\rangle \in L_{s_2}^{D,D'}\}$, hence $\gamma^{D,D'}(s_1) \geq \gamma^{D,D'}(s_2)$.

□

Using both coverage and discriminability measures strikes a balance between the completeness and generality of link specifications. They can be aggregated by harmonic mean, here denoted by $\kappa^{D,D'}$, just like F-measure does with precision and recall (see Figure 3.1).

Definition 12 (Harmonic mean of discriminability and coverage)**.**

$$\kappa^{D,D'}(s) = \frac{2 \times \gamma^{D,D'}(s) \times \delta^{D,D'}(s)}{\gamma^{D,D'}(s) + \delta^{D,D'}(s)}$$

Example 4 (Evaluation measures on link key expressions)**.** *The evaluation of the link key expressions of Example 2 are:*

link key expression	$\delta^{D,D'}$	$\kappa^{D,D'}$	$\gamma^{D,D'}$	$\phi^{D,D'}$
k	*.75*	*.66*	*.58*	*.5*
h	*1.0*	*.67*	*.50*	*.5*
l	*1.0*	*.67*	*.50*	*.5*

The evaluation of all candidate link keys for the data sets of Example 1 is provided in Figure 3.1. Of course, given that the measures are computed on generated links and that h *and* l *generate the same link set, their measures are the same.*

These measures are heuristics which work well under some conditions, especially non supervised, not others. Moreover, they do not evaluate links but *sets of links*, this is the reason why they judge characteristics of these sets of links.

They are primarily used for ranking link specifications: they select among the possible specifications, but do not directly decide on the generated links. Hence, the goal is not to obtain optimal values for the generated link sets: these are highly dependent on the dataset. Their correction with respect to generated links can only be evaluated after hand (like the correction of the chosen measures or weights in other approaches).

Similarly, the respective sizes of the data sets will affect all specifications since they are evaluated against the same pair of data sets. Thus, they have little influence on the extracted link specifications. This effect may be countered by giving, a proportional importance to coverage.

We already showed [Atencia et al. 2014] that, if small modifications are made in the data sets, link key extraction is robust: measures change but their order is preserved.

4.2 Functionality

An alternative for discriminability would be functionality:

Definition 13 (Functionality)**.**

$$\phi^{D,D'}(s) = \frac{|L_{s,\leq 1}^{D,D'}(D)| + |L_{s,\leq 1}^{D,D'}(D')|}{|c^D| + |c'^{D'}|}$$

It has the advantage of always favouring the most specific link specifications:

Property 4. *If* $s_1 \preceq^{D,D'} s_2$*, then* $\phi^{D,D'}(s_1) \leq \phi^{D,D'}(s_2)$

4.3 New non-supervised measures: functionality again

The evaluation criterion provided above did not gave good results on some large data sets. Thus, we introduce two other evaluation measures: the functionality score ($\dot{\phi}$) and the harmonic mean ($\dot{\kappa}$) of functional discriminability and coverage. The functionality score assesses the capacity of a compound link key expression to generate a functional link set. It favours instances that are functionally linked and penalises the others in function of the number of instances they are linked to.

Definition 14 (Functionality score)**.** $\dot{\phi}^{D,D'}(s) =$

$$\frac{\sum_{a \in L_{s,\geq 1}^{D,D'}(D)} 1 - log_2(|\{b|\langle a,b\rangle \in L_s^{D,D'}\}|) + \sum_{b \in L_{s,\geq 1}^{D,D'}(D')} 1 - log_2(|\{a|\langle a,b\rangle \in L_s^{D,D'}\}|)}{2 \times L_s^{D,D'}}$$

Functional precision and recall measure how far a link set is from a bijection between the set of instances c^D and $c'^{D'}$.

Definition 15 (Functional discriminability and coverage)**.**

$$\dot{\delta}^{D,D'}(s) = \frac{|L_{s,=1}^{D,D'}(D)| + |L_{s,=1}^{D,D'}(D')|}{2 \times |L_s^{D,D'}|} \quad \dot{\gamma}^{D,D'}(s) = \max\left(\frac{|L_{s,=1}^{D,D'}(D)|}{|c^D|}, \frac{|L_{s,=1}^{D,D'}(D')|}{|c'^{D'}|}\right)$$

As before, the harmonic mean is used to combine them:

Definition 16 (Harmonic mean of functional discriminability and coverage)**.**

$$\dot{\kappa}^{D,D'}(s) = \frac{2 \times \dot{\gamma}^{D,D'}(s) \times \dot{\delta}^{D,D'}(s)}{\dot{\gamma}^{D,D'}(s) + \dot{\delta}^{D,D'}(s)}$$

Example 5 (Functional evaluation measures on link key expressions)**.** *The functional evaluation of the link key expressions of Example 2 are:*

link key expression	$\dot{\delta}^{D,D'}$	$\dot{\kappa}^{D,D'}$	$\dot{\gamma}^{D,D'}$	$\dot{\phi}^{D,D'}$
k	*.75*	*.60*	*.67*	*.75*
h	*1.0*	*.54*	*.50*	*1.0*
l	*1.0*	*.54*	*.50*	*1.0*

This does not change much with respect to the more classical measures of Example 4. The fact that $\dot{\delta}$ provides the same values as $\dot{\phi}$ is due to objects never linked to more than two objects, otherwise functionality would decrease more.

5. Compound link specifications

By having this common view on link specifications, we can consider them more as complementary than opposed! They are part of the same framework and it is possible to use them jointly.

5.1 Composition operators

Operators are introduced to be applied to form compound link specifications:

Definition 17 (Compound link specifications)**.** *Given two data signatures S and S', a compound link specification over S and S' is:*
- *either an atomic link specification over S and S', or*
- *$s \wedge s'$ (conjunction),*
- *$s \vee s'$ (disjunction),*
- *$s \setminus s'$ (complementation),*
- *$s|s'$ (injective complementation),*
- *$s \boxplus s'$,*
- *$s \oplus s'$ (symmetric injective complementation),*

with s and s', two compound link specifications over S and S'.

We will use $\times$ as a placeholder for any of these operators. Their semantics is provided by the links they generate:

Definition 18 (Link sets generated by compound link specifications)**.** *Let D and D' be two data sets of signatures S and S'. Let s and s' be two link specifications over their signatures, the* link set generated by *a compound link specification is the subset of $c^D \times c'^{D'}$ defined as:*

$$
\begin{aligned}
L^{D,D'}_{s \wedge s'} &= L^{D,D'}_{s} \cap L^{D,D'}_{s'} \\
L^{D,D'}_{s \vee s'} &= L^{D,D'}_{s} \cup L^{D,D'}_{s'} \\
L^{D,D'}_{s \setminus s'} &= L^{D,D'}_{s} \setminus L^{D,D'}_{s'} \\
L^{D,D'}_{s|s'} &= L^{D,D'}_{s} \cup (L^{D,D'}_{s'} \dot{-} L^{D,D'}_{s}) \\
L^{D,D'}_{s \boxplus s'} &= L_{(s|s') \wedge (s'|s)} \\
L^{D,D'}_{s \oplus s'} &= L^{D,D'}_{s \wedge s' | s \vee s'}
\end{aligned}
$$

such that $L^{D,D'}_{s} \dot{-} L^{D,D'}_{s'} = \{\langle o, o' \rangle \in L^{D,D'}_{s}; o \notin \pi(L^{D,D'}_{s'}) \wedge o' \notin \pi'(L^{D,D'}_{s'})\}$.

Conjunction and disjunction are simple ways to combine link specifications. It would have been possible to define a complementation of link specifications acting as 'negation' and to derive one of conjunction or disjunction from the other through De Morgan laws. However, we do not use negation since extracting algorithms focus on finding good link specification candidates, so there is no reason to use them negatively. We only have the opportunity to suppress from the link set generated by a link specification those links which would conflict with another ($\dot{-}$). This is inline with the search for more one-to-one link sets. Figure 5.1 positions the various subsets of L_s and $L_{s'}$ and should help understanding the relations between the composition operators.

Injective complementation $s|s'$ completes the link set generated by s with links generated by s' which do not involve individuals already linked by s. Hence it is less prone to generate

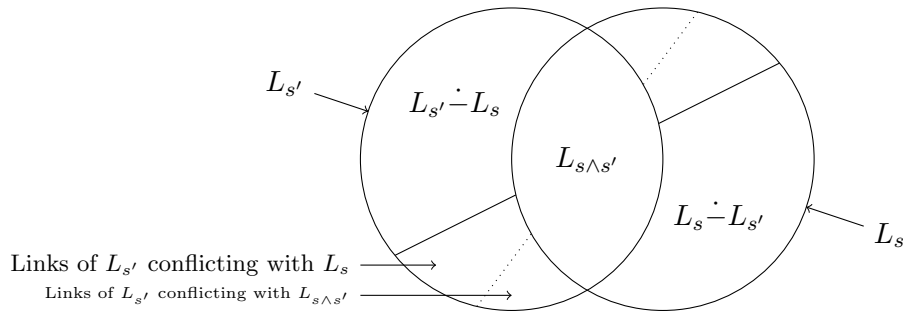

Figure 5.1: Positioning of the various subsets of links when combining. The set $L_{s'}$ can be partitioned into three sets: $L_{s \wedge s'}$, $L_{s'} \dot{-} L_s$ and the links of $L_{s'}$ conflicting with L_s. This last set contains a smaller set: those links of $L_{s'}$ conflicting with $L_{s \wedge s'}$.

a non one-to-one link set. Symmetric injective complementation $s \oplus s'$ preserves all links generated by both link specifications s and s' and injectively complements them with links generated by either s or s'. This operator does not prioritise on the two link specifications, hence it may be less injective than the injective complementation: if two different links involving the same individuals are generated by each of the link specifications independently, then they will be preserved. These operations satisfy useful properties and relations.

Property 5. *$\vee$, $\wedge$, $\boxplus$ and $\oplus$ are commutative; $\vee$ and $\wedge$ are associative.*

Proof. For $\vee$ and $\wedge$, it is a direct consequence of the properties of union and intersection. Commutativity of $\oplus$ comes from that of the two others: $L^{D,D'}_{s \oplus s'} = L^{D,D'}_{s \wedge s' | s \vee s'} = L^{D,D'}_{s' \wedge s | s' \vee s} = L^{D,D'}_{s' \oplus s}$. □

Symmetric injective complementation can be expressed in a different way:

Property 6. $s \oplus s' \simeq ((s \wedge s')|s) \vee ((s \wedge s')|s')$

Proof.

$$\begin{aligned}
L^{D,D'}_{s \oplus s'} &= L^{D,D'}_{s \wedge s' | s \vee s'} \\
&= L^{D,D'}_{s \wedge s'} \cup (L^{D,D'}_{s \vee s'} \dot{-} L^{D,D'}_{s \wedge s'}) \\
&= (L^{D,D'}_{s} \cap L^{D,D'}_{s'}) \cup ((L^{D,D'}_{s} \cup L^{D,D'}_{s'}) \dot{-} (L^{D,D'}_{s} \cap L^{D,D'}_{s'})) \\
&= (L^{D,D'}_{s} \cap L^{D,D'}_{s'}) \cup ((L^{D,D'}_{s} \dot{-} (L^{D,D'}_{s} \cap L^{D,D'}_{s'})) \cup (L^{D,D'}_{s'} \dot{-} (L^{D,D'}_{s} \cap L^{D,D'}_{s'}))) \\
&= L^{D,D'}_{s \wedge s'} \cup ((L^{D,D'}_{s} \dot{-} (L^{D,D'}_{s \wedge s'}) \cup (L^{D,D'}_{s'} \dot{-} L^{D,D'}_{s \wedge s'})) \\
&= (L^{D,D'}_{s \wedge s'} \cup (L^{D,D'}_{s} \dot{-} L^{D,D'}_{s \wedge s'})) \cup (L^{D,D'}_{s \wedge s'} \cup (L^{D,D'}_{s'} \dot{-} L^{D,D'}_{s \wedge s'})) \\
&= L^{D,D'}_{(s \wedge s')|s} \cup L^{D,D'}_{(s \wedge s')|s'} \\
&= L^{D,D'}_{((s \wedge s')|s) \vee ((s \wedge s')|s')}
\end{aligned}$$

□

So far we have defined binary operators between two compound link specifications. They can be applied to a sequence of specifications.

Definition 19 (Uniform compound link specification)**.** *A compound link specification using a single composition operator is called uniform.*

When the composition operator $\times$ of a uniform compound link specification is associative, then we drop the parentheses and write it in the linear form $k_1 \times k_2 \times \ldots k_n$.

5.2 Relations between compound link specifications

The composition operators induce an order on the compound link specifications which can still be defined as anti-monotonic with the inclusion of generated link sets.

Compound link specifications may be compared and some reduced to others:

Property 7. *If $s \preceq s'$, then $s \wedge s' \simeq s'$, $s \vee s' \simeq s \oplus s' \simeq s|s' \simeq s$ and $s \boxplus s' \simeq s'|s$.*

Proof. If $s \preceq s'$, then $L_{s'} \subseteq L_s$, thus $L_{s'} \dot{-} L_s = \varnothing$. Moreover, $L_{s \wedge s'} = L_s \cap L_{s'} = L_{s'}$, $L_{s \vee s'} = L_s \cup L_{s'} = L_s$, and $L_{s \oplus s'} = L_{s \wedge s' | s \vee s'} = L_{s \wedge s'} \cup (L_{s \vee s'} \dot{-} L_{s \wedge s'}) = L_s \cup (L_{s'} \dot{-} L_s) = L_{s|s'}$. But $L_{s|s'} = L_s \cup (L_{s'} \dot{-} L_s) = L_s$. Finally, $L_{s \boxplus s'} = L_{(s|s') \wedge (s'|s)} = [L_s \cup (L_{s'} \dot{-} L_s)] \cap [L_{s'} \cup (L_s \dot{-} L_{s'})]$ $= L_s \cap [L_{s'} \cup (L_s \dot{-} L_{s'})] = (L_s \cap L_{s'}) \cup (L_s \cap (L_s \dot{-} L_{s'})) = L_{s'} \cup (L_s \dot{-} L_{s'}) = L_{s'|s}$. □

Thus, given a set S of link specifications and a set O of combination operators as provided in Definition 17, we can define $Comb(S, O)$ as the set of all link specifications they generate (the closure of S by the operators). $Comb(S, O)$ is ordered by an extension $\preceq$ of the initial specification order such that:

$$s \preceq^{D,D'} s' \text{ iff } L^{D,D'}(s) \supseteq L^{D,D'}(s')$$

The extended subsumption relation ($\preceq$) is straightforwardly extended to compound link specifications. Property 8 states the relations between compound link specifications with respect to $\preceq$.

Property 8. *If s and s' are two link specifications, then*

$$s \vee s' \preceq s|s' \preceq s \preceq s \wedge s' \tag{5.1}$$

$$s|s' \preceq s \boxplus s' \tag{5.2}$$

$$s \vee s' \preceq s \oplus s' \preceq s \boxplus s' \preceq s \wedge s' \tag{5.3}$$

Proof. By Definition 18, $L^{D,D'}_{s \wedge s'} \subseteq L^{D,D'}_s \subseteq L^{D,D'}_{s|s'} \subseteq L^{D,D'}_{s \vee s'}$ this is sufficient to establish Equation 5.1.

Also by Definition 18, $L^{D,D'}_{s \boxplus s'} = L^{D,D'}_{(s|s') \wedge (s'|s)} = L^{D,D'}_{s|s'} \cap L^{D,D'}_{s'|s} \subseteq L^{D,D'}_{s|s'}$, which proves Equation 5.2.

Concerning Equation 5.3, the two extremities are easily proven as follows: $L^{D,D'}_{s \oplus s'} = L^{D,D'}_{s \wedge s'} \cup (L^{D,D'}_{s \vee s'} \dot{-} L^{D,D'}_{s \wedge s'})$, but, $L^{D,D'}_{s \wedge s'} \subseteq L^{D,D'}_{s \vee s'}$ and $L^{D,D'}_{s \vee s'} \dot{-} L^{D,D'}_{s \wedge s'} \subseteq L^{D,D'}_{s \vee s'}$ hence $L^{D,D'}_{s \oplus s'} \subseteq L^{D,D'}_{s \vee s'}$. $L^{D,D'}_{s \boxplus s'} = L^{D,D'}_{(s|s') \wedge (s'|s)} = L^{D,D'}_{s|s'} \cap L^{D,D'}_{s'|s}$, but $L^{D,D'}_{s|s'} \supseteq L^{D,D'}_s$ and $L^{D,D'}_{s'|s} \supseteq L^{D,D'}_{s'}$, hence $L^{D,D'}_{s \boxplus s'} \supseteq L^{D,D'}_{s \wedge s'}$. Finally, the internal equation is obtained by:

$$\begin{aligned}
L^{D,D'}_{s \boxplus s'} &= L^{D,D'}_{(s|s') \wedge (s'|s)} \\
&= L^{D,D'}_{s|s'} \cap L^{D,D'}_{s'|s} \\
&= [L^{D,D'}_s \cup (L^{D,D'}_{s'} \dot{-} L^{D,D'}_s)] \cap [L^{D,D'}_{s'} \cup (L^{D,D'}_s \dot{-} L^{D,D'}_{s'})] \\
&\subseteq [L^{D,D'}_s \cup ((L^{D,D'}_s \cup L^{D,D'}_{s'}) \dot{-} (L^{D,D'}_s \cap L^{D,D'}_{s'}))] \cap [L^{D,D'}_{s'} \cup ((L^{D,D'}_s \cup L^{D,D'}_{s'}) \dot{-} (L^{D,D'}_s \cap L^{D,D'}_{s'}))] \\
&= (L^{D,D'}_s \cap L^{D,D'}_{s'}) \cup ((L^{D,D'}_s \cup L^{D,D'}_{s'}) \dot{-} (L^{D,D'}_s \cap L^{D,D'}_{s'})) \\
&= L^{D,D'}_{s \wedge s'} \cup (L^{D,D'}_{s \vee s'} \dot{-} L^{D,D'}_{s \wedge s'}) \\
&= L^{D,D'}_{s \wedge s' | s \vee s'} \\
&= L^{D,D'}_{s \oplus s'}
\end{aligned}$$

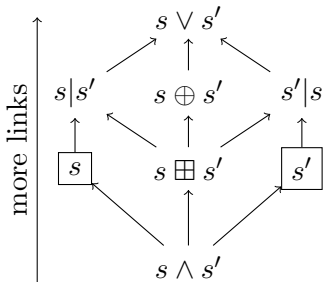

Figure 5.2: Extended subsumption ($\preceq$) between compound link specifications from two link specifications s and s. Edges are subsumption between expresions, such that the outbound node subsumes the inbound one.

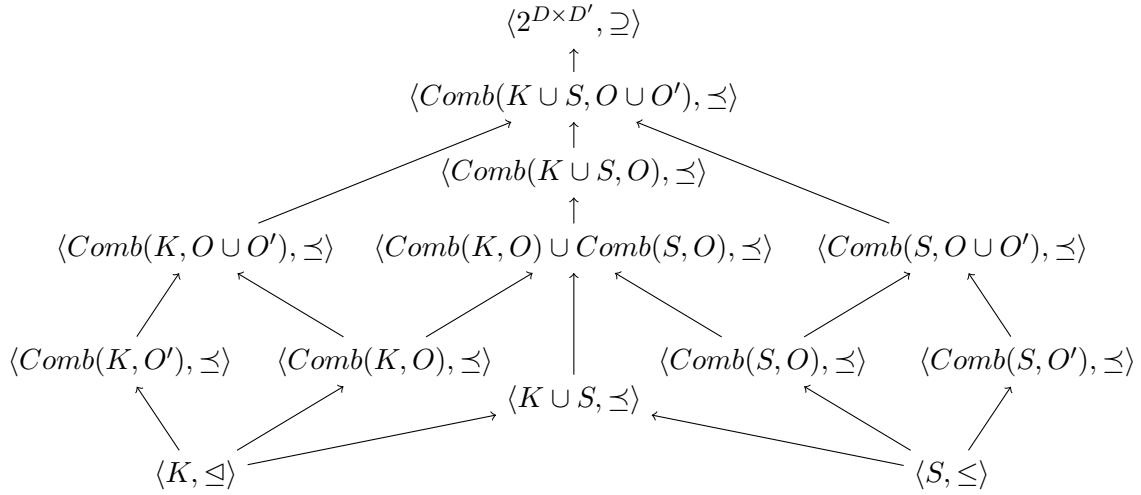

Figure 5.3: Graph of combinations

The second inclusion statement is due to $L_{s'}^{D,D'} \subseteq L_s^{D,D'} \cup L_{s'}^{D,D'}$ and $L_s^{D,D'} \supseteq L_s^{D,D'} \cap L_{s'}^{D,D'}$, hence $L_{s'}^{D,D'} \dot{-} L_s^{D,D'} \subseteq L_s^{D,D'} \cup L_{s'}^{D,D'}) \dot{-} (L_s^{D,D'} \cap L_{s'}^{D,D'})$ on both sides of the $\cap$ sign. $\square$

Figure 5.2 illustrates Property 8 showing the relations between link keys composed from two link key expressions s' and s.

It is possible to define homomorphisms between these partially ordered sets (mapping of the specifications preserving the order), themselves based on homomorphism between link specifications, based on the links that such specifications would generate. This can be represented by the following Figure 5.3 (K is the set of link key candidates or expressions and S the set of —numeric here— specifications):

In fact, this may be considered in other ways:

– We only represented the combination with all the operators, but it is possible to consider those with less operators, and thus the uniform compositions. These would naturally fit in the diagram as orders approximating orders with more elements. The graph above is defined with respect to D and D'. Actually, it is possible to define the lower part, i.e. below the to element, independently from it.

– A specification would be mapped into another if for any D and D' they would necessarily generate the same set of links. Then, the specific structure concerning one pair of data sets may be obtained by a functor projecting specifications generating the same set of links for this data set into the same node.
– Finally, because all relations are based on generated links, it is possible to merge the whole lower part within a single partial order... which is basically what $\langle Comb(K \cap S, O), \preceq_{D,D'} \rangle$ is.

In addition, so far K and S are not really atomic. S's elements may be further analysed, eventually as aggregation of atomic conditions. Consider that an atomic numeric condition is defined by the set N of $\langle p, p', \sigma, \theta \rangle$ such that $\langle p, p' \rangle$ is a pair of properties, σ a similarity function and θ a threshold. A pair of objects $\langle o, o' \rangle$ satisfies the condition if $\sigma(p(o), p'(o')) \geq \theta$ (θ may be 0 if necessary). Consider further that these atomic conditions can be combined by a set $\theta = Aggr \times [0\ 1]$ of aggregators associated with further thresholds. For instance, $\langle Min, .2 \rangle$. Then, $S = Comb(N, \theta)$.

As far as K is concerned, the set of conditions C are well-known as $\forall p, p'$ and $\exists p, p'$. They are simply combined by conjunction. Hence, $K = Comb(C, \wedge)$.

Even more fun, the key conditions may now be further embedded as numeric conditions. $\sigma\forall(v, v') = 1$, if $v = v'$, 0 otherwise, $\sigma_\exists(v, v') = 1$, if $v \cap v' \neq \varnothing$, 0 otherwise, and $\theta\forall = \theta\exists = \theta\exists\forall = 0$. Then $Aggr_K = Average$ and $\theta_K = 1$, make that

$$\begin{aligned} K &= Comb(\forall, \exists \times P_D \times P_{D'}, \wedge) \\ &\approx Comb(P_D \times P_{D'} \times \sigma_\forall, \sigma_\exists \times \theta\exists\forall, \langle Aggr_K, \theta_K \rangle) \subseteq Comb(N, \theta) = S \end{aligned}$$

5.3 Quality measures

5.3.1 Discrimination, coverage and friends

Existing quality measures for single link specifications are defined on their generated links, so they can be directly extended to compound link specifications. Indeed, $\delta^{D,D'}(s)$ and $\gamma^{D,D'}(s)$ only rely on $L_s^{D,D'}$ which is also defined for compound link specifications (Definition 18).

Property 3 still holds with respect to compound link specifications and, in particular, the relations stated in Property 8.

5.3.2 Marginal and residual measures

However, it is interesting to consider the contribution of adding a link specification to a compound link specification. This can be obtained by computing the marginal contribution of the former to the result.

Actually, what we want to measure is:

Definition 20 (Marginal discriminability and coverage)**.**

$$\begin{aligned} \bar{\delta}_{s_1}^{D,D'}(s_2) &= \delta^{D,D'}(s_1 \times s_2) - \delta^{D,D'}(s_1) \\ \bar{\gamma}_{s_1}^{D,D'}(s_2) &= \gamma^{D,D'}(s_1 \times s_2) - \gamma^{D,D'}(s_1) \end{aligned}$$

Such marginal values may be positive or negative. However, for $\vee$ marginal coverage cannot be negative and for $\wedge$ and $\oplus$ marginal discriminability cannot be negative. Depending on the operators, different ways of computing them have to be developed. Even if $\times$ is

$s_1 \times s_2$	$\kappa^{D,D'}(s_1)$	$\bar{\delta}_{s_1}^{D,D'}(s_2)$	$\bar{\kappa}_{s_1}^{D,D'}(s_2)$	$\bar{\gamma}_{s_1}^{D,D'}(s_2)$
$k_2 \vee k_6$	.77	.04	.09	.17
$k_2 \vee k_8$	.77	0.0	0.0	0.0
$k_8 \vee k_4$	.50	0.0	.41	.50
$k_4 \vee k_6$	.67	0.0	.33	.50
$k_8 \vee k_5$	.50	-.55	.10	.59

Table 5.1: Marginal evaluation measures for the compound link keys of Table 6.1.

commutative, this measure is not symmetric, i.e. $\bar{\delta}_{s_1}(s_2) \neq \bar{\delta}_{s_2}(s_1)$. This type of measure is only used to explore the space of link specifications.

The marginal measures can be defined as follows:

Definition 21 (Residual discriminability and coverage)**.**

$$\bar{\delta}_{s_1}^{D,D'}(s_2) = \delta^{D,D'}(s_1) - \frac{min(|\{a|\langle a,b\rangle \in (L_{s_2}^{D,D'} \setminus L_{s_1}^{D,D'}\}|, |\{b|\langle a,b\rangle \in L_{s_2}^{D,D'} \setminus L_{s_1}^{D,D'}\}|)}{|L_{s_1}^{D,D'}|}$$

$$\bar{\gamma}_{s_1}^{D,D'}(s_2) = \gamma^{D,D'}(s_1) - \gamma^{D/s_1,D'/s_1}(s_2)$$

such that: $D/s_1 = \{\langle s,p,o\rangle \in D|\ \nexists i \in I_{D'}; \langle s,i\rangle \in L_{s_1}^{D,D'}\ \text{or}\ \langle o,i\rangle \in L_{s_1}^{D,D'}\}$.

Definition 22 (Marginal discriminability and coverage)**.**

$$\bar{\delta}_{s_1}^{D,D'}(s_2) = \delta^{D,D'}(s_1) - \delta^{D/s_1,D'/s_1}(s_2)$$
$$\bar{\gamma}_{s_1}^{D,D'}(s_2) = \gamma^{D,D'}(s_1) - \gamma^{D/s_1,D'/s_1}(s_2)$$

such that: $D/s_1 = \{\langle s,p,o\rangle \in D|\ \nexists i \in I_{D'}; \langle s,i\rangle \in L_{s_1}^{D,D'}\ \text{or}\ \langle o,i\rangle \in L_{s_1}^{D,D'}\}$.

Table 5.1 provides examples of the marginal measures on the same compound as Table 6.1.

So far we considered as given a sequence of link keys. We now consider how it may be extracted from the data.

6. Application to compound link keys

Here we more specifically consider compound specifications applied to link keys. We first define composition operators for combining link key expressions, we consider the relationships between the resulting compound link key expressions and apply them the evaluation measures.

6.1 Compound link keys

Example 6 applies the composition operators to the previously provided link keys.

Example 6 (Compound link key expressions and generated links)**.** *Taking into account the link key candidates of Figure 3.1, one can compute that:*

$$\begin{aligned} L^{D,D'}_{k_4 \vee k_6} &= \{\langle i_2, z_2\rangle, \langle i_3, z_3\rangle, \langle i_4, z_4\rangle, \langle i_6, z_6\rangle, \langle i_7, z_7\rangle, \langle i_8, z_8\rangle\} \\ L^{D,D'}_{k_7|k_6} &= \{\langle i_7, z_2\rangle, \langle i_6, z_4\rangle, \langle i_2, z_2\rangle, \langle i_3, z_3\rangle, \langle i_4, z_4\rangle, \langle i_8, z_8\rangle\} \\ L^{D,D'}_{k_4 \vee k_6} = L^{D,D'}_{k_6|k_7} &= \{\langle i_2, z_2\rangle, \langle i_3, z_3\rangle, \langle i_4, z_4\rangle, \langle i_6, z_6\rangle, \langle i_7, z_7\rangle, \langle i_8, z_8\rangle\} \\ L^{D,D'}_{k_6 \oplus k_7} &= \{\langle i_2, z_2\rangle, \langle i_3, z_3\rangle, \langle i_4, z_4\rangle, \langle i_8, z_8\rangle\} \\ L^{D,D'}_{k_5 \wedge k_9} = L^{D,D'}_{k_0} &= \{\} \end{aligned}$$

None of these link sets, but arguably the last one, are extensions of a candidate link key (or of a simple link key expression). However, some compound link keys are redundant, e.g. $k_4 \vee k_6$ *and* $k_6|k_7$*.*

The set of uniform compound link key candidates from a set of link key candidates K composed by operator $\times$ is noted $K^{\times}$. We use it quotiented by $\simeq$ as $K^{\times}_{\simeq}$, since $\simeq$ is an equivalence relation.

6.2 Compound link keys and subsumption, meet and join

The links generated by conjunction ($\wedge$) are the same as those generated by join ($\triangledown$). This could make one think that conjunction is redundant. However, the set of link key candidates is not closed by $\triangledown$: if k and h are link key expressions, then, by definition, $k \triangledown h$ is a link key expression, but, if k and h are link key candidates, $k \triangledown h$ is not necessarily a link key candidate. Nevertheless, there is always a link key candidate that generates the same links (i.e. equivalent through $\simeq$):

Property 9. *If* k *and* h *are link key candidates, then there exists a link key candidate* l *such that* $k \wedge h \simeq l$*.*

Proof. By definition of the concept lattice, for each pair of link keys k and h, there exists a greatest common subsumee l such that $l \trianglelefteq k$ and $l \trianglelefteq h$. Either, $l = k \triangledown h$ or $l \neq k \triangledown h$, in which case, $L^{D,D'}_{l} = L^{D,D'}_{k \triangledown h} = L^{D,D'}_{k \wedge h}$. □

Property 9 means that concerning link key candidates, the conjunction does not bring any new link set: an already available link key candidate will generate the same link set. For instance, in Figure 3.1, $k_9 \triangledown k_5$, i.e. the link key expression made of the union of the conditions of k_9 and k_5, is not a link key candidate. Their lowest common subsumer with

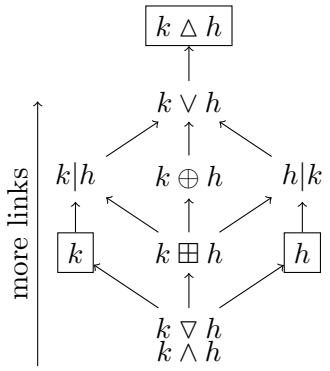

Figure 6.1: Extended subsumption ($\preceq$) between compound link key expressions from two link key expressions k and h. Boxed nodes are link key candidates; edges are subsumption between expresions, such that the outbound node subsumes the inbound one. This figure specialises Figure 5.2 to link key expressions.

respect to $\trianglelefteq$ (or the greatest common subsumee with respect to $\preceq$) is k_0, which is such that $k_9 \triangledown k_5 \trianglelefteq k_0$ and $L_{k_0} = L_{k_9 \wedge k_5}$, i.e. $k_0 \simeq k_9 \wedge k_5$. However, conjunction remains useful to combine compound link key candidates (for which $\triangle$ is not defined).

This property does not apply to the other operators and in particular to disjunction: the links generated by disjunction ($\vee$) are not the same as the links generated by meet ($\triangle$). The set of link key candidates is closed by meet ($\triangle$) [Atencia et al. 2020, Lemma 3], but meet is *not* disjunction ($\vee$) since [Atencia et al. 2020, Lemma 1(2)] $L^{D,D'}_{k \triangle h} \supseteq L^{D,D'}_{k} \cup L^{D,D'}_{h} = L^{D,D'}_{k \vee h}$. This can be checked on Example 6 since $k_3 = k_6 \triangle k_7$ and $L^{D,D'}_{k_3} \supset L^{D,D'}_{k_6} \cup L^{D,D'}_{k_7}$.

The extended subsumption relation ($\preceq$) is straightforwardly extended to compound link keys. Property 10 states the relations between compound link key expressions with respect to $\preceq$.

Property 10. *If s and s' are two link key expressions, then*

$$k \triangle h \preceq k \vee h \preceq k|h \preceq k \preceq k \wedge h = k \triangledown h$$
$$k|h \preceq k \boxplus h$$
$$k \vee h \preceq k \oplus h \preceq k \boxplus h \preceq k \wedge h$$

Proof. From [Atencia et al. 2020, Lemma 1(2)] $L^{D,D'}_{k \triangle h} \supseteq L^{D,D'}_{k} \cup L^{D,D'}_{h} = L^{D,D'}_{k \vee h}$, hence $k \triangle h \preceq k \vee h$. From [Atencia et al. 2020, Lemma 1(3)], $L_{k \triangledown h} = L_k \cap L_h$ which is $L_{k \wedge h}$. So $k \wedge h \simeq k \triangledown h$. The rest was established in Property 8. □

Figure 6.1 illustrates Property 10 by extending Figure 5.2 showing the relations between link keys composed from two link key expressions h and k.

A compound link key expression made of link key candidates is called a compound link key candidate.

6.3 Quality measures

As displayed in the left part of Table 6.1, which shows the values of δ, γ and κ for some of the link key candidates of Figure 3.1, it is already possible to compute them on the compound link key expressions.

$k \times h$	$\kappa^{D,D'}(k)$	$\delta^{D,D'}(k \times h)$	$\kappa^{D,D'}(k \times h)$	$\gamma^{D,D'}(k \times h)$
$k_2 \vee k_6$	.77	.75	.86	1.0
$k_2 \vee k_8$	.77	.71	.77	.83
$k_8 \vee k_4$	.50	1.0	.91	.83
$k_4 \vee k_6$	.67	1.0	1.0	1.0
$k_8 \vee k_5$	.50	.54	.68	.92

Table 6.1: Evaluation measures for some of the compound link keys of Figure 3.1. They all improve coverage (always with $\vee$). All, but the last one, have higher or equal harmonic mean (κ) than the best link key candidates in the lattice ($\kappa(k_2) = .77$). $k_2 \vee k_8$ does not improve on k_2 because k_2 subsumes k_8.

Once the quality of a compound link specification candidate can be measured, the problem is to obtain the best(s) link specification.

7. Extraction strategies for disjunctive link keys

As explained in Section 3.3, the link key candidate extraction process provides a link key candidate lattice, and for each candidate it is possible to compute coverage and discriminability. The question to address is to find a combination of such link key candidates whose overal evaluation measure is higher than these.

From a set K of link key candidates returned by an extraction algorithm, we consider the extraction of the best linear uniform compound link key candidate(s). We concentrate here on the disjunction ($\vee$) operator and will use the harmonic mean of discriminability and coverage (κ) as quality measure.

Such a compound link key candidate will be based on a sequence (or a subset) of link key candidates $K' \subseteq K$. However, if $|K| = n$, there are potentially 2^n linear uniform compound link keys for commutative-associative operators and $\sum_{k=1}^{n} n!/k!$ for non commutative-associative ones. Hence, it is worth determining those of interest more efficiently.

In what follows, we propose three different heuristics to search disjunctions of link key candidates efficiently.

7.1 Extending the best candidates

A natural strategy is to extend the link key candidate that has the best evaluation [Achichi et al. 2016]. However, this is not an optimal strategy.

Property 11. *The best linear uniform compound link key in $K^{\vee}$ with respect to κ does not necessarily contains the best link key candidate.*

Indeed among the candidates of Figure 3.1, the best one is k_2. However, from Figure 7.3, it can be observed that $k_4 \vee k_6$ achieves a harmonic mean of 1.0, that which is not possible for any compound with k_2. This invalidates this strategy.

7.2 Exploring the frontiers

Similarly, it may be possible to start with the candidates with the highest discriminability (k_6, k_8 and k_4) when using $\vee$ or $|$, or the candidate with the highest coverage (k_1 and k_3) when using $\wedge$ or $\oplus$. Let us consider this possibility and first defining these best starting points.

Definition 23 (Frontiers)**.** *Given $\langle K, \preceq \rangle$, $\gamma_\top = \gamma(\top)$ (resp. $\delta_\bot = \delta(\bot)$) with $\top$ (resp. $\bot$) the value of γ (resp. δ) at the supremum (resp. infimum) of the lattice. The upper (resp. lower) frontier $S_\top$ (resp. $S_\bot$) is $\{k \in K \mid \gamma(k) = \gamma_\top \wedge \forall k' \prec k, \gamma(k') < \gamma(k)\}$ (resp. $\{k \in K \mid \delta(k) = \delta_\bot \wedge \forall k' : k \prec k', \delta(k') < \delta(k)\}$). The extended upper (resp. lower) frontier for operator $\times$ $S_\top^\times$ (resp. $S_\bot^\times$) is $\{k \in K^\times \mid \gamma(k) = \gamma_\top \wedge \forall k' \in K^\times : k' \prec k, \gamma(k') < \gamma(k)\}$ (resp. $\{k \in K^\times \mid \delta(k) = \delta_\bot \wedge \forall k' \in K^\times : k \prec k', \delta(k') < \delta(k)\}$).*

By definition, if one link key belongs to both (extended) frontiers, it is optimal because no other link key has a higher disciminability nor higher coverage.

Property 12. *If $S_\top^\times \cap S_\bot^\times \neq \varnothing$, then $S_\top^\times \cap S_\bot^\times$ is the set of best uniform compound link key candidates composed by $\times$ with respect to κ.*

Proof. The elements of $S_\top^\times$ (resp. $S_\bot^\times$) are those link key composed by $\times$ which have the maximal value for γ (resp. δ) on the whole lattice. Hence, if $S_\top^\times \cap S_\bot^\times \neq \varnothing$, its elements are those which have both the highest γ and δ, thus their κ cannot be topped. □

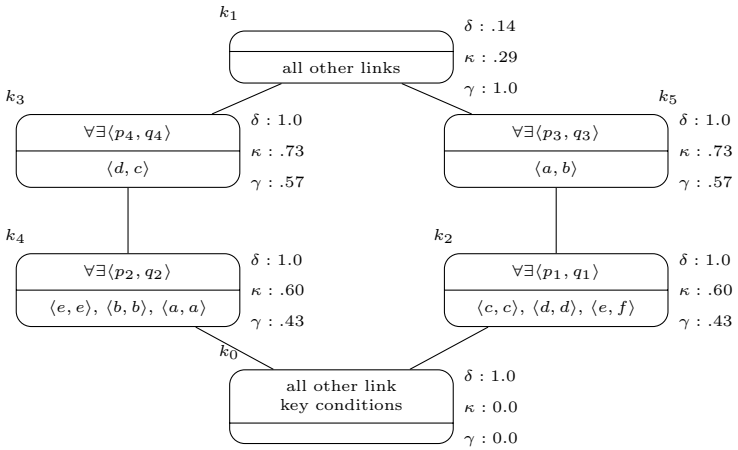

Figure 7.1: Lattice $\langle K, \trianglelefteq\rangle$ of extracted link key candidates from the compound5 linky example (δ=discriminability, γ=coverage, κ=harmonic mean between them).

This is the case of $k_6 \vee k_4$ in Figure 7.3.

In the best case, $S^{\times}_{\top} \cap S^{\times}_{\bot} \neq \varnothing$ and the best (compound) link keys have been obtained. This is what happens on Example 1, and can be seen on Figure 7.3. However, in the worst case, $S_{\top}$ is reduced to the greatest element of the lattice and $S_{\bot}$ to its least element (in this case the complexity of the above algorithm is linear in the size of the lattice). Thus the lattice of compound link keys has to be explored when $S^{\times}_{\top} \cap S^{\times}_{\bot} = \varnothing$.

If the frontiers do not intersect, then the best compound link key candidates are found between the two frontiers. We would like to rely on a property like this one:

Property 13. *The best uniform compound link keys k composed by $\vee$ with respect to κ is such that, there exists $h \in S_{\bot}$ such that $h \preceq k$, and there exists $l \in S_{\top}$ such that $k \preceq l$.*

Unfortunatelly, this property does not hold as shown by Figure 7.1[1]. Here the frontier is $\{k_3, k_5\}$, but the best compound link key is $k_4 \vee k_2$ as can be observed in Figure 7.2.

7.3 Exploiting the antichain lattice

For a given pair of data sets, the link key candidates are organised by $\preceq$ (extended subsumption) in a lattice. The search of disjunctions of link key candidates can be restricted to the search of antichains[2] [Garg 2015] of elements in this lattice (Property 14).

A disjunction of link key candidates can be built from an antichain of link key candidates (given that $\vee$ is associative and commutative).

Property 14. *Any uniform compound link key composed by $\vee$ (resp. $\wedge$) is equivalent to one built from an antichain of the link key candidate lattice.*

Proof. If $k \preceq h$, then $L_{k \vee h} = L_k$ (Property 7), hence $k \vee h \simeq k$. So the uniform compound is equivalent to the disjunction of non comparable link key candidates.

[1] Implemented in linky as compound5.

[2] An *antichain* of a partially ordered set is a subset of pairwise non comparable elements. In our setting, antichains represent non redundant disjunctions of link key candidates. More formally, an antichain is a set of link key candidates $\{k_1, \ldots, k_m\}$ such that for every $i, j = 1, \ldots, m$ with $i \neq j$, neither $k_i \preceq k_j$ nor $k_j \preceq k_i$.

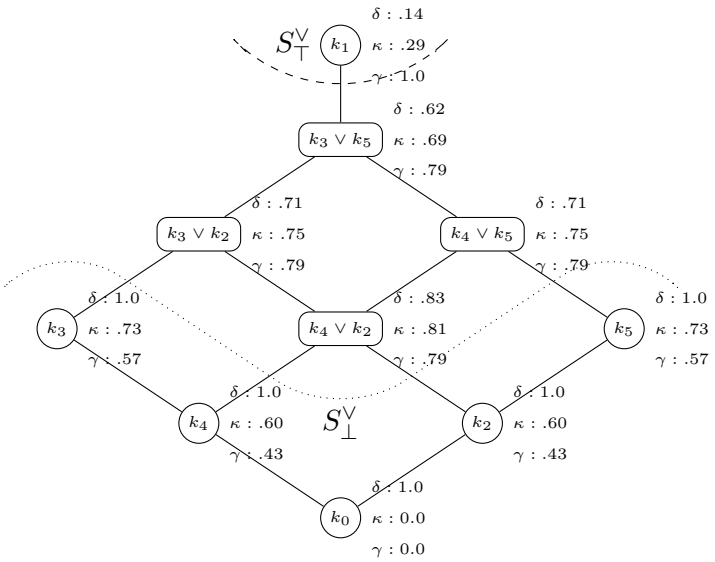

Figure 7.2: Full antichain lattice $\langle K^{\vee}_{\simeq}, \preceq \rangle$ including link keys (in circles) and (maximal) antichains (in rounded boxes) for the link keys of Figure 7.1. The relation is the inclusion of generated link sets. (δ=discriminability, γ=coverage, κ=harmonic mean between them). Below the dotted line, $\delta = 1.0$, above the dashed line $\gamma = 1.0$.

In the same way for conjunction ($\wedge$), if $k \preceq h$, then $L_{k \wedge h} = L_h$ (Property 7), hence $k \wedge h \simeq h$. Hence, for conjunction the set of of link key candidates to compose must also be an antichain, with the additional constraint that it should not contain a pair of link key candidates k and h such that $L_k^{D,D'} \cap L_h^{D,D'} = \varnothing$. However, this is never useful since, by Property 9, they is already an equivalent link key candidate l in the lattice. □

Property 15. *The frontiers are antichains of the link key lattice.*

Proof. This is implicit in Definition 23: all elements in a frontier are non comparable. □

In Example 3.1, these are $\{k_3\}$ and $\{k_4, k_6\}$ which are non maximal antichains.

The number of antichains of a lattice is difficult to establish a priori, the worst case being 2^n. In the case of the link key candidates of Figure 3.1, there are 10 candidates, thus 1024 possible compound. However, there are only 12 maximal antichains covering 30 antichains. All these compound link keys can be organised in a lattice $\langle K^{\vee}_{\simeq}, \preceq \rangle$. The one corresponding to Example 1 is displayed in Figure 7.3: it shows that the procedure of the previous section yields the best compound link key.

Example 7 (Antichains)**.** *Here is the complete list of antichains (first maximal, then non maximal) from the lattice of Figure 3.1. They can be organised as in Figure 7.3.*

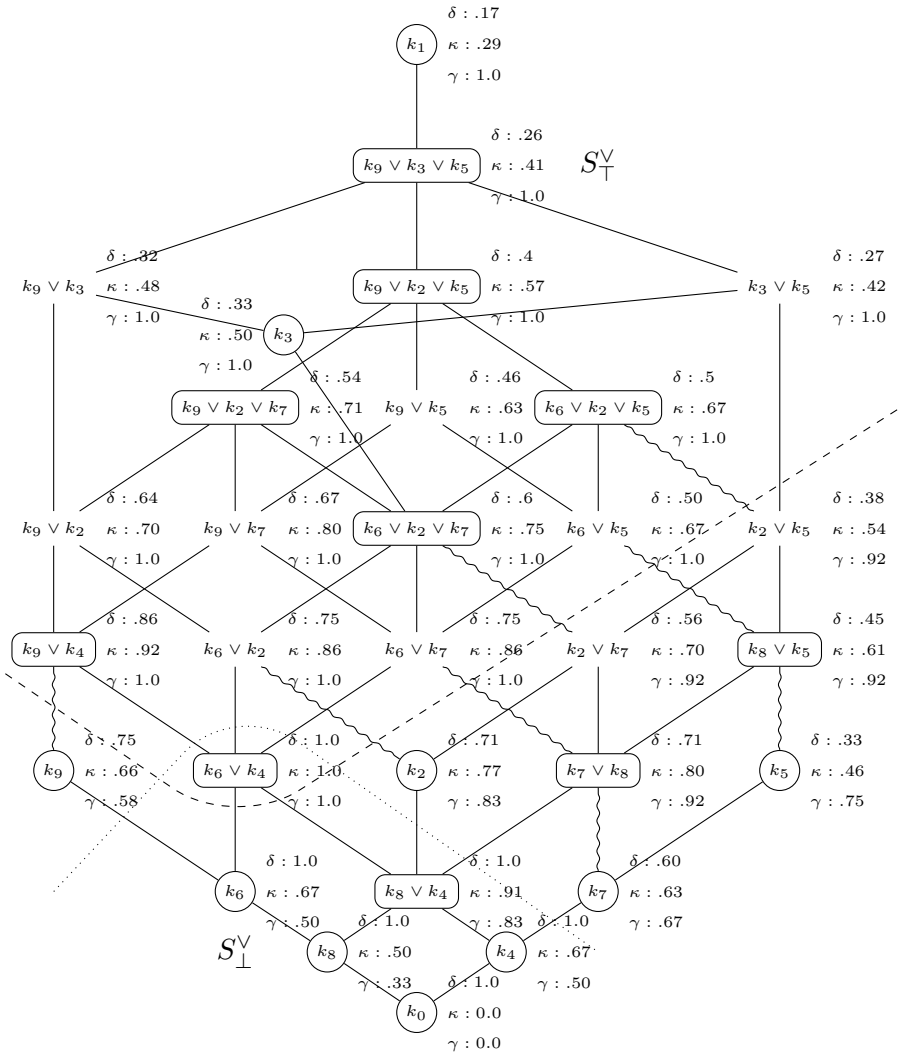

Figure 7.3: Full antichain lattice $\langle K^{\vee}_{\simeq}, \preceq \rangle$ from Example 1 including link keys (in circles), maximal antichains (in rounded boxes) and antichains. (δ=discriminability, γ=coverage, κ=harmonic mean between them). Below the dotted line is $S^{\vee}_{\bot}$, above the dashed line is $S^{\vee}_{\top}$. Ondulating edges represent edges along which discriminability is not decreasing with respect to $\preceq$.

compound link key	$\|L\|$	*linked obj.* D	*linked obj.* D'	*dupl. obj.* D	*dupl. obj.* D'	*measures* γ	κ	δ	ϕ
k_0	0					0.0	0.0	1.0	1.0
k_2	7	23478	23478	48	48	.83	.77	.71	.67
k_3	18	234678	234678	234678	234678	1.0	.50	.33	0.0
k_4	3	234	234			.5	.67	1.0	1.0
k_5	9	234678	234	346	234	.75	.46	.33	.50
k_6	3	678	678			.5	.67	1.0	1.0
k_7	5	23467	234		24	.67	.63	.60	.83
k_8	2	78	78			.33	.50	1.0	1.0
k_9	4	2678	678		8	.58	.66	.75	.92
k_1	36	234678	234678	234678	234678	1.0	.29	.17	0.0
$k_9 \vee k_3 \vee k_5$	23	234678	234678	234678	234678	1.0	.41	.26	0.0
$k_9 \vee k_2 \vee k_5$	15	234678	234678	234678	2348	1.0	.57	.40	.17
$k_9 \vee k_2 \vee k_7$	11	234678	234678	24678	248	1.0	.71	.54	.33
$k_9 \vee k_4$	7	234678	234678	2	8	1.0	.92	.86	.83
$k_6 \vee k_2 \vee k_5$	12	234678	234678	34678	2348	1.0	.67	.50	.25
$k_6 \vee k_2 \vee k_7$	10	234678	234678	4678	248	1.0	.75	.60	.42
$k_8 \vee k_5$	11	234678	23478	34678	234	.92	.61	.45	.33
$k_6 \vee k_4$	6	234678	234678			1.0	1.0	1.0	1.0
$k_7 \vee k_8$	7	234678	23478	7	24	.92	.80	.71	.75
$k_8 \vee k_4$	5	23478	23478			.83	.91	1.0	1.0
$k_9 \vee k_3$	19	234678	234678	234678	234678	1.0	.48	.32	0.0
$k_9 \vee k_5$	13	234678	234678	234678	2348	1.0	.63	.46	.17
$k_3 \vee k_5$	22	234678	234678	234678	234678	1.0	.42	.27	0.0
$k_9 \vee k_2$	11	234678	234678	248	48	1.0	.70	.54	.58
$k_9 \vee k_7$	9	234678	234678	267	248	1.0	.80	.67	.50
$k_6 \vee k_5$	12	234678	234678	34678	234	1.0	.67	.50	.33
$k_2 \vee k_5$	13	234678	23478	34678	2384	.92	.54	.38	.25
$k_6 \vee k_2$	8	234678	234678	48	48	1.0	.86	.75	.67
$k_6 \vee k_7$	8	234678	234678	67	24	1.0	.86	.75	.67
$k_2 \vee k_7$	9	234678	23478	478	248	.92	.70	.56	.50

Exploring the antichain lattice may be achieved in various ways, but it reaches its limits fast if there are many link key candidates. Indeed, (1) the number of non maximal antichains may be daunting, and (2) the best candidate in terms of κ may be anywhere in the lattice. Hence, we consider next how to deal with such problems in large data sets.

7.4 In search of the "best" antichain

We propose three strategies for searching the best antichains. These approaches are not exhaustive but use heuristics that may help to find good antichains.

The *support-driven* strategy filters properties before link key candidate extraction. Properties having a support less than a given threshold, i.e. that are instantiated for less than a given proportion of instances, may be filtered out. This allows to reduce the number of generated link key candidates by the initial algorithm [Atencia et al. 2014] and then to perform an exhaustive enumeration of antichains. This assumes that the best antichains are not composed of link key candidate with properties having a low support.

The *top-k strategy* selects the top k candidates according to some evaluation measure and then performs an exhaustive enumeration of antichains on this selection. This assumes that the best antichains are those which only contain the best link key candidates.

The *expand-best strategy* performs a best-first search. It explores the antichains from the best atomic ones and by iteratively replacing the best antichain by its expansion (i.e. the set of antichains obtained by adding another link key candidate). At each step, an antichain is selected only if it is better than those explored thus far. The process stops after x iterations without improvement. It assumes that the better an antichain is, the more chances that it can produce better antichains.

The two latter approaches, with the data sets of Example 1, return quickly the best disjunction of link keys: $k_6 \vee k_4$. We compare them experimentally in Chapter 8.

8. Experimental evaluation

In this section, we report on experiments with two of the strategies for extracting disjunctions of link key candidates proposed in Chapter 7. Our aim was to test if disjunctions of link key candidates generate better link sets, in terms of F-measure, than single link key candidates.

8.1 Data sets

To test the above hypothesis, we used OAEI data sets and two data sets from the library domain. The data sets of each OAEI task share the same ontology, but this is not the case of the two data sets from the library domain.

We have used available standard benchmarks and refrained from providing our own. Although some of these tests may be favorable to link key extraction, they may also be favorable to other systems, e.g. by having the same ontology on both datasets. It could be noted that, for instance, for Doremus in 2017, the best system (Legato) is from the benchmark provider, the second one is only run on this test. So, we do not compare our results with the results of other data interlinking approaches, as it is not the goal of this evaluation. Besides, the results of the different OAEI campaigns are available online[1].

For each data set, we present a synthetic table with the number of properties (named properties, inverse or composition of properties) that appear in at least one extracted candidate and the number of extracted link key candidates, and the precision, recall and F-measure of the candidate with the highest κ value.

8.2 Methodology

For each OAEI task, we first generated single link key candidates with *Linkex*, our link key candidate extraction tool[2] [Abbas et al. 2019]. We set up Linkex to perform a basic normalisation of data values, and to deal with inverse and 2-length composition of properties. The data normalisation consisted in removing diacritics, tokenising strings and sorting the resulting bag of tokens.

Then, we applied the top-k and expand-best strategies for extracting the best disjunctions of link key candidates according to κ. For the top-k strategy, we chose $k = 10, 15, 20, 25, 30$. Since all these different values of k produced very similar results, we only present here the results of $k = 10$. For expand-best, we stopped the search after $x = 100$ iterations without finding a better antichain.

Finally, the comparison between disjunctions and single link key candidates was done by computing precision, recall and F-measure against the provided reference link sets. For each data set, a table provides the results of the antichain extraction and the precision, recall and F-measure of the antichains with the highest κ value obtained by the two strategies. For the top-10 strategy, it also shows the number of antichains extracted and the number of maximal antichains among them (e.g. 43 and 12 for Restaurants, respectively). For the expand-best strategy, it shows the number of generated antichains — *tested* column (e.g. 337 for Restaurants) — and also the position of the best antichain in the sequence of generated antichains — *best* column (e.g. 34 for Restaurants).

[1] http://oaei.ontologymatching.org
[2] https://gitlab.inria.fr/moex/linkex

Data set	# instances	# properties	# triples	# candidates	Prec.	F-meas.	Rec.
Restaurants1	113	4	1 130	20	0.477	0.58	0.741
Restaurants2	752	4	7 520				
Person11	500	9	9 000	613	1	0.974	0.95
Person12	500	10	7 000				
Person21	600	9	10 800	521	0.206	0.27	0.39
Person22	400	10	5 600				

Table 8.1: OAEI 2010 data sets and extracted link key statistics.

Task	Strategy	Prec.	F-meas.	Rec.	time	#a.c /tested	#max a.c /best
Restaurants	top-10	0.483	0.596	0.777	¡1”	43	12
	expand-best	0.481	0.594	0.777	¡1”	337	34
Person1	top-10	1	1	1	¡1”	223	5
	expand-best	1	1	1	¡3”	1041	901
Person2	top-10	0.348	0.425	0.545	¡1”	311	8
	expand-best	0.265	0.369	0.608	¡3”	30 110	18 523

Table 8.2: Results on OAEI 2010 datasets.

8.3 Simple data sets (OAEI 2010)

Table 8.1 displays the charateristics of the OAEI 2010 data sets[3]: Person1, Person2, Restaurants. Since Restaurant1 has a low number of candidate link keys (18), we performed an exhaustive enumeration of antichains instead of top-10.

For all of them, the two strategies allowed to find an antichain with better F-measure than the best individual link key. The top-10 strategy performed better than the expand-best strategy on Restaurant and Person2. The expand-best strategy generated more antichains than the top-10 strategy. This was specially true for Person2 on which the expand-best strategy generated a large number of antichains. Moreover, the F-measure was particularly low. This is due to the fact that the δ measure is not well-adapted to this specific task since the first data set of Person2 contains a lot of redundancy. All these results are obtained in less than one second.

8.4 Doremus (OAEI 2016)

The Doremus data sets[4] of OAEI 2016 are small data sets — PP-n and BnF-n ($n = 1, 2, 3$), see Table 8.3 — from cultural institutions with different kinds of heterogeneity.

The added value of compound link key expressions is clearly demonstrated in Table 8.4 since all different strategies allow to gain at least 4 points of F-measure with respect to the best atomic link key.

Unlike the previous tasks (Section 8.3), expand-best outperformed top-10 on the three tasks. But, as for the previous tasks, expand-best generated more antichains than top-10. With the top-10 strategy, the size of compound link key expressions are respectively 3 and

[3] http://oaei.ontologymatching.org/2010/
[4] http://islab.di.unimi.it/content/im_oaei/2016/

Data set	# instances	# properties	# triples	# candidates	Prec.	F-meas.	Rec.
PP-1	32	9	2 530	27	0.833	0.714	0.625
BnF-1	32	9	2 189				
PP-2	201	13	12 757	101	0.833	0.712	0.622
BnF-2	201	15	10 622				
PP-3	41	11	2 970	38	0.622	0.571	0.683
BnF-3	41	12	2 610				

Table 8.3: Doremus data sets and extracted link key statistics.

Task	Strategy	Prec.	F-meas.	Rec.	time	#a.c /tested	#max a.c /best
Doremus 1	top-10	0.793	0.754	0.719	¡1”	72	9
	expand-best	0.806	0.794	0.781	¡1”	326	54
Doremus 2	top-10	0.829	0.799	0.771	¡1”	219	9
	expand-best	0.830	0.802	0.776	¡1”	2187	420
Doremus 3	top-10	0.569	0.667	0.805	¡1”	140	9
	expand-best	0.596	0.694	0.829	¡1”	416	82

Table 8.4: Results on Doremus data sets.

4. The expand-best strategy generates larger link key expressions respectively composed of 3 and 9 atomic link keys. This latter strategy favours recall.

8.5 SPIMBench (OAEI 2018)

We also applied these techniques to the SPIMBench Sandbox data set[5] from OAEI 2018. These data sets (Abox1, Abox2) include around 380 instances and 10000 triples. The goal of the task is to find links between instances of classes NewsItem, BlogPost and Programme (they share the same ontologies but our algorithms do not take this into account).

In this case, the number of extracted link key candidates was quite large. This is due to the high number of properties of the data sets (see Table 8.5). Thus, an exhaustive search of all antichains was not feasible.

The best link key candidate is already of a high quality ($\kappa \approx 0.8$). As Table 8.6 shows, none of the two strategies was able to find a better compound link key expression than the best atomic one.

Due to the heuristic nature of the proposed strategies, it is unclear whether this result is the consequence of the non existence of a better link key or the incompleteness of the procedure.

[5] http://oaei.ontologymatching.org/2018/spimbench.html

Data set	# instances	# properties	# triples	# candidates	Prec.	F-meas.	Rec.
Abox1	349	38	10 001	2 277	0.816	0.794	0.773
Abox2	284	58	10 022				

Table 8.5: SPIMBench Sandbox data sets and extracted link key statistics.

Task	Strategy	Prec.	F-meas.	Rec.	time	#a.c /tested	#max a.c /best
SPIMBench	top-10	0.816	0.794	0.773	4”	47	12
	expand-best	0.805	0.788	0.773	1’20”	26 557	3 318

Table 8.6: Results of SPIMBench Sandbox.

Data set	# instances	# properties	# triples	# candidates	Prec.	F-meas.	Rec.
Abes	15 421	6	66 610	933	0.656	0.614	0.578
BnF	8 162	9	106 224				

Table 8.7: Library data sets and extracted link key statistics.

In order to verify the scalability to bigger data sets, we also experimented with the SPIMBench Mainbox data sets, which contain, respectively, 1759 and 1399 instances (no reference is available, hence we cannot evaluate them). 1122 link key candidates were extracted. The top-10 strategy took 2s and the expand-best strategy 3m43s (28153 antichains were evaluated).

8.6 Large data set / Libraries

Finally, we used sample data sets provided by two French libraries: the “Bibliothèque Nationale de France” (BnF),[6] and the “Agence Bibliographique de l’Enseignement Supérieur” (Abes).[7] The sampling consisted in extracting within each data set the authors that have one of the top-1000 most common homonym names (name and first name). The books written by each author are available. The classes to link are the ones representing authors. Unlike the OAEI data sets, these data sets use different ontologies. Only a partial reference link set was available. Table 8.7 provides the statistics of these data sets.

The top-10 strategy marginally improved the F-measure of the best link key candidate, while expand-best had a lower score. Both strategies improved recall but precision was negatively impacted. This may be due to the fact that the results were evaluated against a partial reference. The best disjunction generated by top-10 contains 2 link keys while the one generated by expand-best contains 62 atomic link keys. This last disjunction contains many link keys with very low coverage: 10 only generate one link and 36 generate less than ten links. This goes against the idea that link keys are general linking conditions and that they always cover most of the instances to link. Table 8.8 provides the characteristics of the best extracted disjunctive candidate for the two strategies

[6] https://data.bnf.fr/
[7] https://www.idref.fr/

Task	Strategy	Prec.	F-meas.	Rec.	time	#a.c /tested	#max a.c /best
Libraries	top-10	0.563	0.616	0.679	< 1”	134	16
	expand-best	0.363	0.474	0.681	42”	65112	35193

Table 8.8: Results of the Libraries dataset.

8.7 Conclusion

Overall, our hypothesis was confirmed: disjunctions of link keys bring an improvement to data interlinking with respect to single link keys.

The experimental results show that the top-10 strategy always allows to find a disjunction better than the best single link key candidate. In addition, the expand-best strategy always generates longer disjunctions than the top-10 strategy. Indeed, whereas the top-10 strategy generated disjunctions of only two or three link key candidates in all cases, the expand-best strategy generated very long disjunctions in some cases: 22 for Person2 and 10 for the Doremus2. Consequently, the expand-best strategy favours recall over precision. Furthermore, top-10 scales better than expand-best.

The proposed κ measure is not optimal in the sense that some generated link key candidates are better in terms of F-measure than those selected by κ. This is especially true if the data sets are very different in size (number of instances) and when the target link set is far from a one-to-one mapping. Further work will be needed for identifying more suitable measures.

Concerning the OAEI tasks, we must admit that link keys (single or disjunction candidates) do not provide the best results of the campaigns on the first three tasks. However, these data sets should be considered as easier to be worked with by the other existing data interlinking approaches, as they use the same ontologies, which is not a requirement for link keys. Indeed, Linkex always considers as different the properties of two different datasets. Only the pair of classes whose instances are to be linked is given by input.

9. Conclusion

Link keys are an interesting tool in the data interlinking toolbox. Their extraction is fast, non supervised, alignment-independent, so worth trying. However, we doubt that there exists one single best technique for data interlinking, especially non supervised. There may be cases in which link keys are not suited and a similarity measure will give better results. The converse may occur as well. Combining link specifications may provide benefits none of the two approaches provides alone.

We have defined a framework in which link specifications in general can be combined with composition operators. The semantics of these operators has been given in terms of generated links and evaluation measures naturally extend to these. We have explored the relations between composition operations applied to link keys and the relations between link key conditions.

This work then focussed on link key composition and more specifically disjunction of link keys. We have extended link key candidate extraction to deal with disjunctions of link key candidates. Given the size of the search space, we introduced heuristics to extract disjunctions of link key candidates.

The experimental evaluation shows that the extracted disjunctions improve the F-measure of individual link key candidates. However, the experiments demonstrate that further improvements are needed, either by combining both heuristics, or by developing more suitable unsupervised measures. Indeed, the measures proposed in [Atencia et al. 2014] may not be the best one, at least on compound link keys. Designing better ones may improve the extraction process.

It would be interesting to specify the compound link key extraction problem directly within formal concept analysis and compare the equivalence and benefits of the proposed approach. Can this be directly extracted with formal concept analysis techniques? The answer is yes: the space of intents, instead of that of link keys become that of compound link keys (or even link specifications). The space of extents is still characterised by the set of links they generate. This is a bit more complex than extending the set of condition expressions. Can this be dealt with in RCA? Although this remain to be proved, there is no a priori reason why it could not. Finally, this may also be approached with pattern structure (we have an ordered extent space).

10. Bibliography

Abbas, Nacira, Jérôme David, and Amedeo Napoli (2019). "Linkex: A tool for link key discovery based on pattern structures". en. In: *Proc. ICFCA workshop on Applications and tools of formal concept analysis, Frankfurt (DE)*, pp. 33–38 (cit. on p. 31).

Achichi, Manel, Mohamed Ben Ellefi, Danai Symeonidou, and Konstantin Todorov (2016). "Automatic Key Selection for Data Linking". In: *Proc. 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW), Bologna (IT)*. Vol. 10024. Lecture Notes in Computer Science. Springer, pp. 3–18 (cit. on pp. 6, 7, 25).

Al-Bakri, Mustafa, Manuel Atencia, Jérôme David, Steffen Lalande, and Marie-Christine Rousset (2016). "Uncertainty-sensitive reasoning for inferring sameAs facts in linked data". In: *Proc. 22nd european conference on artificial intelligence (ECAI), Der Haague (NL)*, pp. 698–706 (cit. on p. 6).

Al-Bakri, Mustafa, Manuel Atencia, Steffen Lalande, and Marie-Christine Rousset (2015). "Inferring same-as facts from Linked Data: an iterative import-by-query approach". In: *Proc. 29th AAAI Conference on Artificial Intelligence, Austin (TX US)*. AAAI Press, pp. 9–15 (cit. on p. 6).

Atencia, Manuel, Jérôme David, and Jérôme Euzenat (2014). "Data interlinking through robust linkkey extraction". In: *Proc. 21st European Conference on Artificial Intelligence (ECAI)*. IOS Press, pp. 15–20 (cit. on pp. 5–7, 11, 13, 14, 29, 36).

Atencia, Manuel, Jérôme David, and Jérôme Euzenat (2021). "On the relation between keys and link keys for data interlinking". In: *Semantic web journal* 12.4, pp. 547–567 (cit. on p. 7).

Atencia, Manuel, Jérôme David, Jérôme Euzenat, Amedeo Napoli, and Jérémy Vizzini (2020). "Link key candidate extraction with relational concept analysis". In: *Discrete applied mathematics* 273, pp. 2–20 (cit. on pp. 5, 10, 11, 23).

Atencia, Manuel, Jérôme David, and François Scharffe (2012). "Keys and pseudo-keys detection for web datasets cleansing and interlinking". en. In: *Proc. 18th international conference on knowledge engineering and knowledge management (EKAW), Galway (IE)*. Vol. 7605. Lecture Notes in Computer Science. Springer, pp. 144–153 (cit. on p. 6).

Atencia, Manuel, Jérôme David, and Jérôme Euzenat (2019). "Several link keys are better than one, or extracting disjunctions of link key candidates". en. In: *Proc. 10^{th} ACM international conference on knowledge capture (K-Cap), Marina del Rey (CA US)*, pp. 61–68 (cit. on pp. 2, 9).

Brickley, Dan and R.V. Guha (2014). *RDF Schema 1.1*. Recommendation. https://www.w3.org/TR/rdf-schema/. W3C (cit. on p. 8).

Christen, Peter (2012). *Data Matching—Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer, Heidelberg (DE) (cit. on p. 6).

Euzenat, Jérôme and Pavel Shvaiko (2013). *Ontology matching*. en. 2nd. Heidelberg (DE): Springer. 520 pp. (cit. on pp. 6, 7).

Euzenat, Jérôme, Manuel Atencia, Jérôme David, Amedeo Napoli, and Jérémy Vizzini (2019). *Candidate link key extraction with formal concept analysis*. Deliverable 1.1. Elker project (cit. on p. 11).

Farah, Houssameddine, Danai Symeonidou, and Konstantin Todorov (2017). "KeyRanker: Automatic RDF Key Ranking for Data Linking". In: *Proc. Knowledge Capture Conference (K-CAP), Austin (TX US)*, 7:1–7:8 (cit. on pp. 6, 7).

Ferrara, Alfio, Andriy Nikolov, and François Scharffe (2011). "Data Linking for the Semantic Web". In: *International Journal of Semantic Web and Information Systems* 7.3, pp. 46–76 (cit. on pp. 5, 6).

Ganter, Bernhard and Rudolf Wille (1999). *Formal Concept Analysis: mathematical foundations*. Berlin (DE): Springer (cit. on pp. 11, 12).

Garg, Vijay (2015). *Introduction to lattice theory with computer science applications*. London, UK): John Wiley and sons (cit. on p. 26).

Hogan, Aidan, Antoine Zimmermann, Jürgen Umbrich, Axel Polleres, and Stefan Decker (2012). "Scalable and distributed methods for entity matching, consolidation and disambiguation over linked data corpora". In: *Journal of Web Semantics* 10, pp. 76–110 (cit. on p. 6).

Isele, Robert and Christian Bizer (2013). "Active learning of expressive linkage rules using genetic programming". In: *Journal of web semantics* 23, pp. 2–15 (cit. on p. 6).

Motik, Boris, Peter F. Patel-Schneider, and Bijan Parsia (2012). *OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax (2nd edition)*. Recommendation. https://www.w3.org/TR/owl2-syntax/. W3C (cit. on p. 8).

Nentwig, Markus, Michael Hartung, Axel-Cyrille Ngonga Ngomo, and Erhard Rahm (2017). "A survey of current Link Discovery frameworks". In: *Semantic Web* 8.3, pp. 419–436. DOI: 10.3233/SW-150210 (cit. on pp. 5, 6).

Ngonga Ngomo, Axel-Cyrille and Sören Auer (2011). "LIMES: A Time-Efficient Approach for Large-Scale Link Discovery on the Web of Data". In: *Proc. 22nd International Joint Conference on Artificial Intelligence (IJCAI), Barcelona (ES)*. Barcelona (ES), pp. 2312–2317 (cit. on p. 6).

Saïs, Fatiha, Nathalie Pernelle, and Marie-Christine Rousset (2007). "L2R: A Logical Method for Reference Reconciliation". In: *Proc. 22nd National Conference on Artificial Intelligence (AAAI), Vancouver (CA)*. AAAI Press, pp. 329–334 (cit. on p. 6).

Sherif, Mohamed Ahmed, Axel-Cyrille Ngonga Ngomo, and Jens Lehmann (2017). "Wombat - A Generalization Approach for Automatic Link Discovery". In: *Proc. 14th European semantic web conference (ESWC), Portorož (SL)*. Vol. 10249. Lecture Notes in Computer Science. Springer, pp. 103–119 (cit. on p. 6).

Symeonidou, Danai, Vincent Armant, Nathalie Pernelle, and Fatiha Saïs (2014). "SAKey: Scalable Almost Key Discovery in RDF Data". In: *Proc. 13th International Semantic Web Conference (ISWC), Riva del Garda (IT)*. Vol. 8796. Lecture Notes in Computer Science. Springer, pp. 33–49 (cit. on pp. 6, 7).

Volz, Julius, Christian Bizer, Martin Gaedke, and Georgi Kobilarov (2009). "Silk – A Link Discovery Framework for the Web of Data". In: *Proc. WWW Workshop on Linked Data on the Web, LDOW, Madrid (SP)*. Vol. 538. CEUR Workshop Proceedings. CEUR-WS.org (cit. on p. 6).