Discovery of link keys in resource description framework datasets based on pattern structures

Nacira Abbas, Alexandre Bazin, Jérôme David, Amedeo Napoli

Discovery of Link Keys in Resource Description Framework Datasets Based on Pattern Structures

Nacira Abbas[a], Alexandre Bazin[b], Jérôme David[c], Amedeo Napoli[a]

[a] *Université de Lorraine, CNRS, Inria, LORIA, Nancy, 54000, France*
[b] *Université de Montpellier, CNRS, LIRMM, Montpellier, 34095, France*
[c] *Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, Grenoble, 38000, France*

Abstract

In this paper, we present a detailed and complete study on data interlinking and the discovery of identity links between two RDF –Resource Description Framework– datasets over the web of data. Data interlinking is the task of discovering identity links between individuals across datasets. Link keys are constructions based on pairs of properties and classes that can be considered as rules allowing to infer identity links between subjects in two RDF datasets. Here we investigate how FCA –Formal Concept Analysis– and its extensions are well adapted to investigate and to support the discovery of link keys. Indeed plain FCA allows to discover the so-called link key candidates, while a specific pattern structure allows to associate a pair of classes with every candidate. Different link key candidates can generate sets of identity links between individuals that can be considered as equal when they are regarded as partitions of the identity relation and thus involving a kind of redundancy. In this paper, such a redundancy is deeply studied thanks to partition pattern structures. In particular, experiments are proposed where it is shown that redundancy of link key candidates while not significant when based on identity of partitions appears to be much more significant when based on similarity.

Keywords: Data Interlinking, Link Key Discovery, Link Key Candidate, Formal Concept Analysis, Pattern Structures, Redundancy of Link Sets.

1. Introduction

In this paper, we are interested in data interlinking through the discovery of identity links between two RDF (Resource Description Framework) datasets over the web of data [1]. RDF triples such as (subject,property,object) are the basic bricks of the web of data [2]. Elements in such triples may be described thanks to an RDF Schema and/or OWL –Web Ontology Language– ontologies. Moreover, a given subject can be represented in two different datasets by different individuals. Then data interlinking is the task of discovering identity links between individuals across different datasets. In particular, data interlinking is meaningful in data preparation, data cleaning, and noise reduction.

There are several approaches for discovering identity links between two RDF datasets. Some approaches, as in the systems LIMES [3] and SILK [4], are based on link specifications which are rules declaring whether two elements should be linked. In addition, link specifications may also be specified by users or learned from data [5, 6, 7]. Other approaches are based on the discovery of sufficient conditions for declaring that two subjects are identical [8, 9, 10]. Following this way, *link keys* extend the notion of keys in databases and can be viewed as rules allowing to infer identity links between subjects in two RDF datasets [11]. A link key is based on two sets of property pairs associated with a pair of classes. The property pairs correspond to sufficient conditions for declaring that two subjects are identical w.r.t. the associated pair of classes.

For example, the link key ({(designation,title)},{(creator,author)},(Book,Novel)) states that whenever an instance a of the class Book has the same values for the property designation as an instance b of the class Novel for the property title, and that a and b share at least one value for the properties creator and author, then a and b denote the same entity.

More formally, the link keys which are under investigation in this paper are composed of two sets of property pairs $(\{(p_i, q_i)\}_{i \in I}, \{(p'_j, q'_j)\}_{j \in J})$ associated with a pair of classes (c_1, c_2). Then, whenever an instance a of class c_1 has the same non empty set of values as an instance b of class c_2, i.e. $p_i(a) = q_i(b)$ for all property pairs in $\{(p_i, q_i)\}_{i \in I}$, and shares at least one value for all property pairs in $\{(p'_j, q'_j)\}_{j \in J}$, i.e. $p'_j(a) \cap q'_j(b) \neq \emptyset$, then a and b denote the same entity. The first property set $\{(p_i, q_i)\}_{i \in I}$ corresponds to a "universal quantification" while the second property set $\{(p'_j, q'_j)\}_{j \in J}$ corresponds to an "existential quantification". An identity link is then materialized by the a owl:sameAs b link, where the OWL construction owl:sameAs represents an equivalence relation.

Link keys are not provided with RDF datasets and some algorithms were designed for automatically discovering such link keys [11, 12, 13]. These algorithms try to reduce the search space instead of checking every combination of property pairs and class pairs. They focus on the discovery of "link key candidates" which are made precise below and which involve maximality and closure. Accordingly, straightforward connections were established in [14, 12] between link key discovery and Formal Concept Analysis (FCA [15]). Following the lines proposed in [14], FCA is applied in [12] to a binary table where rows correspond to individual pairs and columns to property pairs. The intent of a resulting concept corresponds to a link key candidate while the extent includes potential identity links between individuals.

Afterwards, these identity links should be validated thanks to appropriate quality measures such as "coverage" and "discriminability" [11]. Coverage depends on linkage between entities, i.e., the more subjects are linked by a candidate the better is the coverage of the candidate. By contrast, discriminability assumes that subjects have to be distinct in each dataset and then measures the capability of a link key candidate to discriminate between subjects.

A generalization of the FCA-based approach was proposed in [13], which relies on pattern structures [16, 17] and which takes into account different pairs of classes at the same time in the discovery of link keys. Link key candidates are searched in one pass and the class pairs associated with each discovered link key candidate are specified. There is no need to iterate over every pair of classes and the discovery process does not require any prior alignment. Moreover pairs of class expressions such as conjunctions or disjunctions of named classes may be returned with link key candidates rather than pairs of named classes.

Actually, link key candidates allow to generate different and maximal link sets, i.e., the mapping between individuals is close to a "bijection". However, some candidates are generating the same link set when the latter is considered as a partition w.r.t. the equivalence relation owl:sameAs. This leads to a form of "redundancy" as two different candidates that are inducing the same partition can be considered as equivalent w.r.t. the generated link sets. This redundancy can be detected thanks to the equivalence classes generated by the owl:sameAs relation. Then two candidates relying on the same partition can be considered as redundant and "merged". It should be noticed that no link key is removed in this case, and that merging records redundancy. In this way, the objective of the research work presented in [18] was to check the existence and to evaluate the significance of such redundancy.

Based on "partition pattern structures" (PPS) introduced in [19] for discovering functional dependencies in datasets, we were able to formalize link key discovery in taking into account sets of equivalent individuals w.r.t. owl:sameAs partitions. Then, a pattern concept represents a link key candidate and the related partition induced by the candidate. Experiments provided three main results: (i) the redundancy of link keys in real-world datasets is not so significant, (ii) the discovery of link key candidates based on PPS is efficient and returns whenever possible a reduced number of non redundant candidates, (iii) the quality of the returned link keys is quite high when compared to competitors.

This paper is based on two previous publications at the "International Conference on Concept Lattices and Applications" (CLA 2020 and 2022) [13, 18]. It proposes a complete and revised synthesis of both papers including an alternative formalization, additional explanations, and extended experiments. The application of partition pattern structure for detecting redundancy in link key discovery and an alternative way of dealing with redundancy are made precise. For completing and generalizing the research work initiated in [18], we have carried out experiments where the equality operation corresponding to the owl:sameAs relation is weakened and replaced with a similarity operation. These experiments tell us that redundancy based on equality is in general small and not so important. By contrast, redundancy based on similarity appears to be much more significant with a similarity above rather high thresholds. In particular, this shows that this research line brings some benefits and contributes in a substantial way to general research about link keys.

The summary of the paper is as follows. Section 2 introduces RDF datasets and the basics about link key expressions/candidates, and link key discovery. Then in Section 3 the formalism of pattern structures, which extends plain Formal Concept Analysis, is made precise and supports the discovery of link key candidates with pairs of classes. Examples of a pattern structure, a pattern concept lattice, and a set of link key candidates are given in Section 3. In

Section 5 redundancy of link key candidates is discussed, and as well the way how partition pattern structures allow to detect such redundancy. Section 6 provides details about experiments, describes the datasets which are reused, and shows how the use of a similarity brings another point of view on redundancy of link key candidates. Finally, before Conclusion, Section 7 presents the state of the art and especially the approaches which are close to the present research work.

2. Basics About Link Keys

The process of discovering link keys takes as input two RDF datasets, computes a set of potential link keys, called link key candidates, and evaluates them thanks to quality measures. This section introduces different definitions about RDF datasets that are used in the following.

2.1. RDF Dataset

An RDF dataset represents data as a collection of graphs associating entities to values or to other entities through properties. For the sake of simplicity and without loss of generality, we do not make the distinction between "RDF graph" and an "RDF dataset" in this paper.

Definition 1 (RDF Dataset)**.** Let U denote a set of IRIs, i.e., "Internationalized Resource Identifiers", B a set of blank nodes, i.e., "anonymous resources" or "variables", and L a set of literals, i.e., "string values".

An RDF dataset is a set of triples $(s, p, o) \in (U \cup B) \times U \times (U \cup B \cup L)$.

Here after, D denotes an RDF dataset, and (s, p, o) a triple with a subject $s \in (U \cup B)$, a predicate $p \in U$, and an object $o \in (U \cup B \cup L)$. Then $S(D) = \{s \mid \exists p, o\ (s, p, o) \in D\}$ denotes the set of subject identifiers, while $P(D) = \{p \mid \exists s, o\ (s, p, o) \in D\}$ and the set of property identifiers. In addition, $p(s) = \{o \mid (s, p, o) \in D\}$ denotes the set of objects –or values– associated with s through property p.

Furthermore, $NC(D) = \{c \in U \mid \exists s\ (s, \mathsf{rdf{:}type}, c) \in D\}$ denotes the set of named classes, where rdf:type stands for the instantiation link in RDF, and $I_{NC}(c) = \{s \mid \exists s\ (s, \mathsf{rdf{:}type}, c) \in D\}$ denotes the set of instances of a named classes $c \in NC(D)$.

Finally, an identity link is an RDF triple such as $(a, \mathsf{owl{:}sameAs}, b)$, stating that the IRIs a and b are referring to the same entity, or a and b are denoting the same individual.

Example 1. In Figure 1, $\mathsf{P(D_1)}$ and $\mathsf{P(D_2)}$ are sets of property identifiers:

$\mathsf{P(D_1)} = \{$cAge, cFN, cLN, given, famN, date, desig, creat$\}$,

$\mathsf{P(D_2)} = \{$pAge, pLN, pFN, first, name, year, title, auth$\}$.

$\mathsf{NC(D_1)}$ and $\mathsf{NC(D_2)}$ are sets of named classes over $\mathsf{D_1}$ and $\mathsf{D_2}$:

$\mathsf{NC(D_1)} = \{$Character, Woman, Scientist, Book$\}$,

$\mathsf{NC(D_2)} = \{$Persona, FemScientist, Dictionary, Novel$\}$.

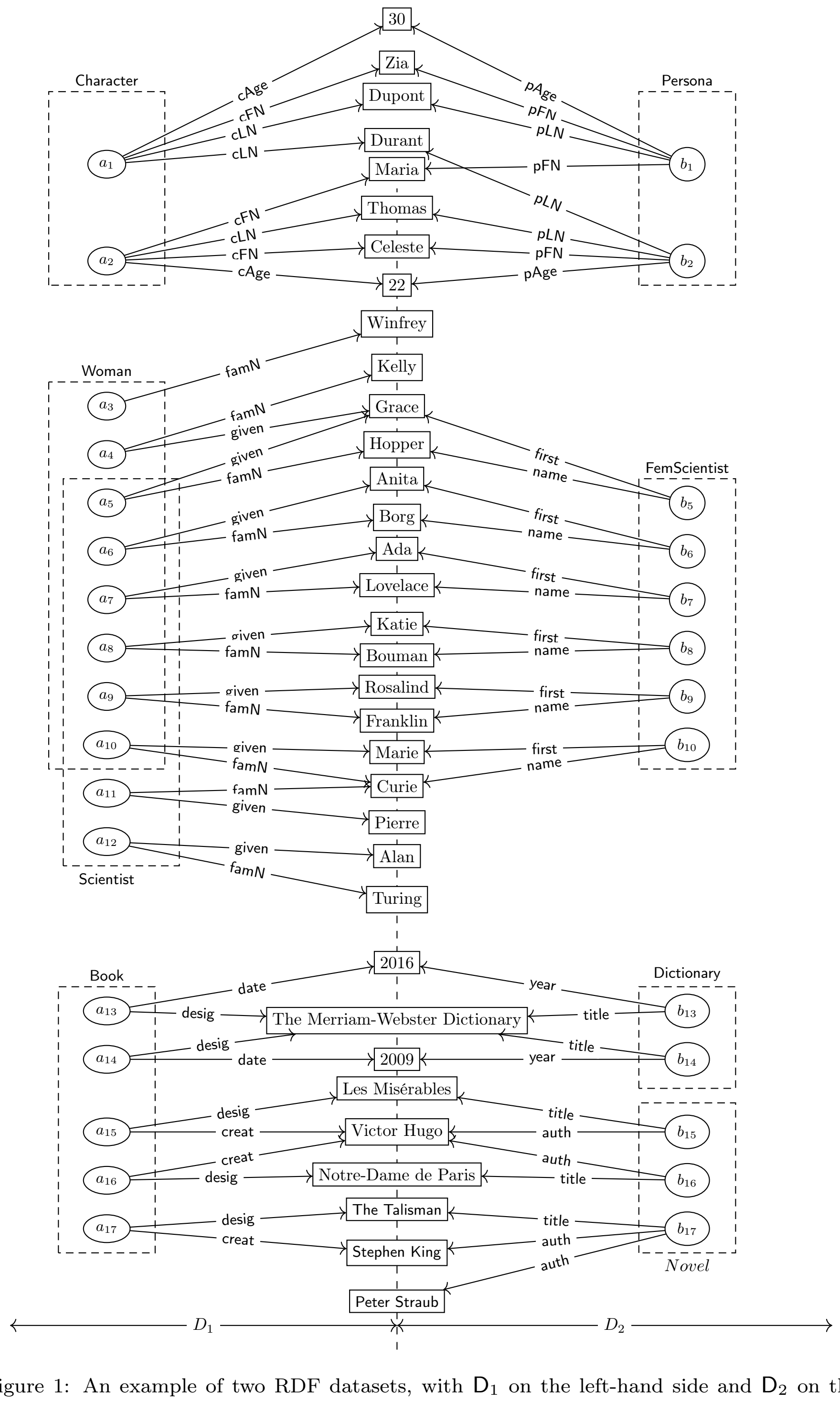

Figure 1: An example of two RDF datasets, with D_1 on the left-hand side and D_2 on the right-hand.

There are also different sets of instances namely:

$\mathsf{I}_{\mathsf{NC}}(\mathsf{Character}) = \{\mathsf{a}_1, \mathsf{a}_2\}$, $\mathsf{I}_{\mathsf{NC}}(\mathsf{Woman}) = \{\mathsf{a}_3, \ldots, \mathsf{a}_{10}\}$, $\mathsf{I}_{\mathsf{NC}}(\mathsf{Book}) = \{\mathsf{a}_{13}, \ldots, \mathsf{a}_{17}\}$, $\mathsf{I}_{\mathsf{NC}}(\mathsf{Scientist}) = \{\mathsf{a}_5, \ldots, \mathsf{a}_{12}\}$, $\mathsf{I}_{\mathsf{NC}}(\mathsf{Persona}) = \{\mathsf{b}_1, \mathsf{b}_2\}$, $\mathsf{I}_{\mathsf{NC}}(\mathsf{Dictionary}) = \{\mathsf{b}_{13}, \mathsf{b}_{14}\}$, $\mathsf{I}_{\mathsf{NC}}(\mathsf{FemScientist}) = \{\mathsf{b}_5, \ldots, \mathsf{b}_{10}\}$, and $\mathsf{I}_{\mathsf{NC}}(\mathsf{Novel}) = \{\mathsf{b}_{15}, \mathsf{b}_{16}, \mathsf{b}_{17}\}$.

Furthermore, the "value" of b_{17} for property auth is $\mathsf{auth}(\mathsf{b}_{17}) = \{\text{Stephen King}, \text{Peter Straub}\}$.

It is possible to lift the above restriction to named classes and to consider conjunctions of named classes such as $c_1 \sqcap_{class} c_2$, or disjunctions of named classes such as $c_1 \sqcup_{class} c_2$. The intuition justifying such an extension relies on the fact that ontologies related to RDF datasets are usually based on different levels of abstraction. For example, the set of female scientists may be represented by the conjunction of the classes "Woman" and "Scientist" in one dataset, and by the named class "FemScientist" in another dataset. In such a case, it is interesting to be able to take into account conjunctions and also disjunctions of classes. Then in agreement with the corresponding constructions in Description Logics [20], the set of class expressions over the RDF dataset D denoted by $C(D)$, including named classes, plus conjunctions ($c_1 \sqcap_{class} c_2 \in C(D)$) and disjunctions of named classes ($c_1 \sqcup_{class} c_2 \in C(D)$), can be recursively defined as follows:

– If $c \in NC(D)$ then $c \in C(D)$,

– if $c_1, c_2 \in C(D)$ then $c_1 \sqcap_{class} c_2 \in C(D)$,

– if $c_1, c_2 \in C(D)$ then $c_1 \sqcup_{class} c_2 \in C(D)$.

Accordingly, the set of instances of a class $c \in C(D)$, denoted by $I(c)$, is recursively defined as follows:

– If $c \in NC(D)$ then $I(c) = I_{NC}(c)$,

– if $c_1, c_2, c_3 \in C(D)$ such as $c_1 = c_2 \sqcap_{class} c_3$, then $I(c_1) = I(c_2) \cap I(c_3)$,

– if $c_1, c_2, c_3 \in C(D)$ such as $c_1 = c_2 \sqcup_{class} c_3$, then $I(c_1) = I(c_2) \cup I(c_3)$.

2.2. Link key expressions, generated links and candidate link keys

Link keys are based on a syntax and a semantics, and they are used to generate identity links between individuals. The discovery of link keys relies on the definitions of "link key expression", "identity link", and "link key candidate", as made precise here below.

Definition 2 (Link key expression and generated identity link set)**.** A *link key expression* over two RDF datasets D_1 and D_2 is a tuple $k = (Eq, In, (c_1, c_2))$ such that $In \subseteq P(D_1) \times P(D_2)$, $Eq \subseteq In$, $c_1 \in C(D_1)$, and $c_2 \in C(D_2)$.

The set of identity links generated by k is denoted by $L(k)$ and includes the set of pairs of instances $(a, b) \in I(c_1) \times I(c_2)$ satisfying:

(i) for all $(p, q) \in Eq$, $p(a) = q(b)$ and $p(a) \neq \emptyset$,

(ii) for all $(p, q) \in In \setminus Eq$, $p(a) \cap q(b) \neq \emptyset$.

Intuitively, a link key expression $k = (Eq, In, (c_1, c_2))$ generates the identity link (a, b) between individuals a instance of class c_1 and b instance of class c_2, when (i) for all $(p, q) \in Eq$, a has the same values for p as b has for q, and (ii) for all $(p', q') \in In$, a and b share at least one value for p' and q'.

The number of link key expressions may be exponential w.r.t. the number of properties. In addition, it is possible that several link key expressions generate the same set of identity links. Then, to reduce the number of potential link keys, link key discovery algorithms only consider the so-called *link key candidates*. The latter correspond to link key expressions that generate at least one link and that are "maximal" among all link key expressions generating the same links.

Definition 3 (Link key candidate)**.** A *link key candidate* is a link key expression $k_1 = (Eq_1, In_1, (c_1, c_2))$ such that:
(i) $L(k_1) \neq \emptyset$,
(ii) there does not exist another link key expression $k_2 = (Eq_2, In_2, (c_1, c_2))$ over D_1 and D_2 such that $Eq_1 \subset Eq_2$, $In_1 \subset In_2$, and $L(k_1) = L(k_2)$.

For example, consider the expressions k_1 and k_2:
k_1 = ({(date, year),(desig, title)},{(date, year),(desig, title)}, (Book, Dictionary)),
k_2 = ({(date, year)},{(date, year)},(Book, Dictionary)).

The link key expressions k_1 and k_2 both generate the same set of links, namely $\mathsf{L}(\mathsf{k}_1) = \mathsf{L}(\mathsf{k}_2)$ = {(a_{13},b_{13}), (a_{14},b_{14})}. As k_1 is larger than k_2, i.e. {(date,year)} $\subset$ {(date,year),(desig,title)}, and as there is no other link key expression generating the same set of links as k_1, then k_1 is considered as a link key candidate.

2.3. Quality measures

A link key candidate can be promoted as a "valid link key" as soon as it generates only valid identity links. As it is not feasible to check one by one every identity link, adapted measures have been introduced for evaluating the quality of a link key candidate. In this way, "discriminability" and "coverage" are two quality measures that work in an unsupervised scenario, i.e., when reference links are not available, as this is the case most of the time [11].

Intuitively, coverage (resp. discriminability) evaluates how close is a set of links to a surjective (resp. injective) map. Hence, coverage is maximum when all instances of the considered classes are linked to "at least" another instance. Discriminability is maximum when instances are linked to "at most" one instance.

Definition 4. Let $L \subseteq I(c_1) \times I(c_2)$ be a set of links between classes c_1 and c_2. Let us denote $\pi_1(L) = \{a | (a, b) \in L\}$ and $\pi_2(L) = \{b | (a, b) \in L\}$.

The *coverage* of L over c_1 and c_2 is defined as:

$$co(L, c_1, c_2) = \frac{|\pi_1(L) \cup \pi_2(L)|}{|I(c_1) \cup I(c_2)|}$$

The *discriminability* of L is defined as:

$$di(L) = \frac{min(|\pi_1(L)|, |\pi_2(L)|)}{|L|}$$

Similarly to the F-measure, discriminability and coverage can be combined into a harmonic mean denoted by hm ($hmean$).

2.4. A Semantics for Link Keys

Link keys are not only syntactic constructions and a semantics may be defined for such expressions. Usually, the semantics of link keys reuses the Description Logics (DL) for interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ [20]. The set $\Delta^{\mathcal{I}}$ is called the interpretation domain and is composed of individuals, while the interpretation function $\cdot^{\mathcal{I}}$ maps conceptual expressions into subsets of the interpretation domain, i.e., subsets of individuals. Such a semantics allows to check the consistency of knowledge bases related by link keys, to infer subsumption relations between link keys, and to define the validity of a link key w.r.t. an interpretation.

Definition 5 (Link key)**.** A *link key* is an expression of the form $(Eq, In, (c_1, c_2))$ where Eq and In are sets of pairs of properties with $Eq \subseteq In$, and c_1 and c_2 are classes.

An interpretation $\mathcal{I}$ satisfies a link key $(Eq, In, (c_1, c_2))$ if, for any $\alpha \in c_1^{\mathcal{I}}$ and $\beta \in c_2^{\mathcal{I}}$ verifying (1) and (2) below it can be inferred that $\alpha = \beta$.
(1) $p_i^{\mathcal{I}}(\alpha) = q_i^{\mathcal{I}}(\beta)$ with $p_i^{\mathcal{I}}(\alpha) \neq \emptyset$ for all $(p_i, q_i) \in Eq$,
(2) $p_j^{\mathcal{I}}(\alpha) \cap q_j^{\mathcal{I}}(\beta) \neq \emptyset$ for all $(p_j, q_j) \in In$,

The link keys which are considered here are a merge of "weak eq-link keys" and "weak in-link keys" as this is introduced and made precise in [21]. However, the present work is not directly related to semantics of link keys and we will no more elaborate in this direction.

3. Basics About Pattern Structures

Pattern structures [16, 17] are an extension of Formal Concept Analysis [15] aimed at adapting the FCA formalism to objects whose descriptions is too complex to be expressed as a set of binary attributes. This is the case when objects have attributes whose values can be numbers, intervals, sequences, trees, graphs... It should be noticed that "object" here denotes an element or an individual which has a description and not as a component of an RDF triple.

Definition 6 (Pattern structure)**.** A *pattern structure* is a triple $(\mathcal{G}, (\mathcal{D}, \sqcap), \delta)$, where $\mathcal{G}$ is a set of *objects*, $\mathcal{D}$ is a set of *descriptions*, and $\delta\colon \mathcal{G} \mapsto \mathcal{D}$ maps an object to its description.

Moreover, the operator $\sqcap$ is a *similarity operator* that defines the similarity between two descriptions. In addition, given two descriptions d_1 and d_2, $\sqcap$ is associated with a partial ordering denoted by $\sqsubseteq$ as follows: $d_1 \sqcap d_2 = d_1$ iff $d_1 \sqsubseteq d_2$.

In particular, the set of potential descriptions $(\mathcal{D}, \sqsubseteq)$ forms a meet-semilattice. The semi-lattice $(\mathcal{D}, \sqsubseteq)$ can be interpreted as the set of descriptions ordered by the $\sqsubseteq$ relation (subsumption). Actually $\sqcap$ maps pairs of descriptions to their least common subsumer as explained in [16]. For example, Figure 2 depicts a simple pattern structure in which objects are described by intervals. Details and examples about "interval pattern structures" can be found in [17].

Objects	Descriptions
g_1	$[1, 2]$
g_2	$[2, 3]$
g_3	$[3, 4]$

Figure 2: A pattern structure with three objects g_1, g_2 and g_3 whose description is an interval. The similarity $\sqcap$ between two intervals is defined by their convex hull, i.e. $[a, b] \sqcap [c, d] = [min(a, c), max(b, d)]$. Then $\delta(g_1) \sqcap \delta(g_2) = [1, 2] \sqcap [2, 3] = [1, 3]$.

As in plain FCA, given a pattern structure $(\mathcal{G}, (\mathcal{D}, \sqcap), \delta)$, two derivation operators denoted by $\cdot^\diamond$ can be defined as follows:

$$\cdot^\diamond : 2^\mathcal{G} \mapsto \mathcal{D}, G^\diamond = \bigsqcap_{g \in G} \delta(g)$$

$$\cdot^\diamond : \mathcal{D} \mapsto 2^\mathcal{G}, d^\diamond = \{g \in \mathcal{G} \mid d \sqcap \delta(g) = d\}$$

It should be noticed that, alternatively, $d \sqcap \delta(g) = d$ can be rewritten as $d \sqsubseteq \delta(g)$. Given a set of objects G, $G^\diamond$ returns the most specific description common to all the objects in G. Dually, given a description d, $d^\diamond$ returns all the objects whose description is more specific than d. Based on these two derivation operators we can define a pattern concept as:

Definition 7 (Pattern concept)**.** A *pattern concept* in $(\mathcal{G}, (\mathcal{D}, \sqcap), \delta)$ is a pair (A, d) with $A \subseteq \mathcal{G}$ and $d \in \mathcal{D}$ such that $A = d^\diamond$ and $d = A^\diamond$.

For example, in Figure 3, $(\{g_1, g_2\}, [1, 3])$ is a concept as $\{g_1, g_2\}^\diamond = \delta(g_1) \sqcap \delta(g_2) = [1, 3]$ and dually $[1, 3]^\diamond = \{g_1, g_2\}$. Then, as this is the case in plain FCA, the set of all concepts of a pattern structure can be partially ordered by inclusion of extents to form a complete lattice called the "pattern concept lattice" of the pattern structure.

In the pattern concept lattice, it can be checked that every object is attached to a unique concept, i.e., the corresponding "object pattern concept" (the "object concept" in plain FCA) [15]. Dually, for every description there is an "description pattern concept", also called an "attribute pattern concept" (the "attribute concept" in plain FCA). Intuitively, considering the order from top to bottom in the lattice, as soon as a description is attached to a given pattern concept, it is attached to all the pattern concepts which are below. For example, the description pattern concept of interval $[2, 4]$ is the pattern concept $(\{g_2, g_3\}, [2, 4])$ and interval $[2, 4]$ is subsumed by the intents $[2, 3]$ and $[3, 4]$ lying in the pattern concepts below. Recall that for descriptions d_1 and d_2, $d_1 \sqcap d_2 = d_1$ iff $d_1 \sqsubseteq d_2$.

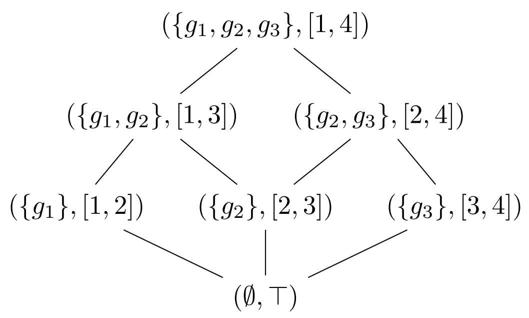

Figure 3: The pattern concept lattice related to the pattern structure given in Figure 2. The $\top$ symbol is arbitrarily added to the meet-semilattice of intervals in order to make it a lattice.

This can be read for intervals as follows: $[2,3] \sqcap [2,4] = [2,4]$ iff $[2,4] \subseteq [2,3]$ [17]. Dually, considering the order from bottom to top in the lattice, as soon as an object is attached to a given pattern concept, it is attached to all the pattern concepts which are above. For example, the object pattern concept of $\{g_1\}$ is the pattern concept $(\{g_1\}, [1,2])$, and $\{g_1\}$ is lying in all extents above this pattern concept.

More formally, the object pattern concept is defined as $\gamma(g) = (g^{\diamond\diamond}, g^{\diamond})$, while the description pattern concept is defined as $\mu(d) = (d^{\diamond}, d^{\diamond\diamond})$. Moreover, the set of object pattern concepts ordered by the inclusion relation on the extents is called the *OC-poset* of the pattern structure [22]. Actually, this is a partially ordered set and not necessarily a lattice. More details and an example involving the OC-poset is proposed in Section 5.

4. Discovering Link Keys Within Pattern Structures

In this section, we present the use of pattern structures to discover link key candidates in RDF datasets. The pairs of classes considered in the link key expressions are restricted to class expressions composed of named classes, or conjunctions of named classes such as $c_1 \sqcap_{class} c_2$, or disjunctions of named classes such as $c_1 \sqcup_{class} c_2$. Recall that conjunctions and disjunctions of named classes correspond to respectively conjunction and disjunction of concepts in Description Logics [20].

Definition 8 (*LK*-pattern structure)**.** Let D_1 and D_2 be two RDF datasets. The *LK*-pattern structure supporting link key candidate discovery over D_1 and D_2 is defined as the triple $(S(D_1) \times S(D_2), (E, \sqcap), \delta)$ where:

*(*i) Objects: $S(D_1) \times S(D_2)$ denotes the set of subjects pairs over D_1 and D_2.

*(*ii) Descriptions and similarity operator ($\sqcap$): E denotes the set of link key expressions $k = (Eq, In, (c_1, c_2))$ over D_1 and D_2, where $c_1 \in C(D_1)$ and $c_2 \in C(D_2)$.

The similarity operator ($\sqcap$) is such that:

Objects	**Descriptions**		
$\mathsf{S}(\mathsf{D}_1) \times \mathsf{S}(\mathsf{D}_2)$	Eq	In	$\mathsf{C}(\mathsf{D}_1)\times \mathsf{C}(\mathsf{D}_2)$
(a_1,b_1)	{(cAge,pAge)}	{(cFN,pFN), (cLN,pLN)}	(Character, Persona)
(a_1,b_2)	{}	{(cLN,pLN)}	(Character, Persona)
(a_2,b_1)	{}	{(cFN,pFN)}	(Character, Persona)
(a_2,b_2)	{(cAge,pAge)}	{(cFN,pFN), (cLN,pLN)}	(Character, Persona)
(a_4,b_5)	{(given,first)}	{(given,first)}	(Woman, FemScientist)
(a_5,b_5)	{(given,first), (famN,name)}	{(given,first), (famN,name)}	(Woman$\sqcap_{class}$Scientist, FemScientist)
(a_6,b_6)	{(given,first), (famN,name)}	{(given,first), (famN,name)}	(Woman$\sqcap_{class}$Scientist, FemScientist)
(a_7,b_7)	{(given,first), (famN,name)}	{(given,first), (famN,name)}	(Woman$\sqcap_{class}$Scientist, FemScientist)
(a_8,b_8)	{(given,first), (famN,name)}	{(given,first), (famN,name)}	(Woman$\sqcap_{class}$Scientist, FemScientist)
(a_9,b_9)	{(given,first), (famN,name)}	{(given,first), (famN,name)}	(Woman$\sqcap_{class}$Scientist, FemScientist)
(a_{10},b_{10})	{(given,first), (famN,name)}	{(given,first), (famN,name)}	(Woman$\sqcap_{class}$Scientist, FemScientist)
(a_{11},b_{10})	{(famN,name)}	{(famN,name)}	(Scientist, FemScientist)
(a_{13},b_{13})	{(date,year), (desig,title)}	{(date,year), (desig,title)}	(Book,Dictionary)
(a_{13},b_{14})	{(desig,title)}	{(desig,title)}	(Book,Dictionary)
(a_{14},b_{13})	{(desig,title)}	{(desig,title)}	(Book,Dictionary)
(a_{14},b_{14})	{(date,year), (desig,title)}	{(date,year), (desig,title)}	(Book,Dictionary)
(a_{15},b_{15})	{(creat,auth), (desig,title)}	{(creat,auth), (desig,title)}	(Book,Novel)
(a_{15},b_{16})	{(creat,auth)}	{(creat,auth)}	(Book,Novel)
(a_{16},b_{15})	{(creat,auth)}	{(creat,auth)}	(Book,Novel)
(a_{16},b_{16})	{(creat,auth), (desig,title)}	{(creat,auth), (desig,title)}	(Book,Novel)
(a_{17},b_{17})	{(desig,title)}	{(creat,auth), (desig,title)}	(Book,Novel)

Table 1: The LK-pattern structure corresponding to the datasets D_1 and D_2 given in Figure 1.

$(Eq_1, In_1, (c_1^{D_1}, c_1^{D_2})) \sqcap (Eq_2, In_2, (c_2^{D_1}, c_2^{D_2})) =$
$(Eq_1 \cap Eq_2, In_1 \cap In_2, (c_1^{D_1} \sqcup_{class} c_2^{D_1}, c_1^{D_2} \sqcup_{class} c_2^{D_2}))$.

(iii) Mapping δ:
$\delta : S(D_1) \times S(D_2) \longrightarrow E$ associates a pair $(s_1, s_2) \in S(D_1) \times S(D_2)$ to its description $\delta((s_1, s_2)) = (Eq, In, (c_1, c_2))$ where:
$Eq = \{(p_1, p_2) \mid p_1(s_1) = p_2(s_2) \neq \emptyset\}$,
$In = \{(p_1, p_2) \mid p_1(s_1) \cap p_2(s_2) \neq \emptyset\}$,
c_1 is the conjunction of the named classes to which s_1 belongs,
i.e., $c_1 = \bigsqcap_{i \in [1,\ell_1]} cl_i$ where cl_i verifies (s_1,rdf:type,cl_i) for $i \in [1, \ell_1]$,
similarly, c_2 is the conjunction of the named classes to which s_2 belongs,
i.e., $c_2 = \bigsqcap_{j \in [1,\ell_2]} cl_j$ where (s_2,rdf:type,cl_j) for $j \in [1, \ell_2]$.

For example, the LK-pattern structure corresponding to the datasets given in Figure 1 is detailed in Table 1. The pattern concept lattice corresponding to this LK-pattern structure is displayed in Figure 4.

In the pattern concept lattice, every pattern concept is based on a pair of closed sets, namely an intent representing a link key candidate and an extent representing the related link set, plus a pair of classes related to the type of the subjects under study. For example, the link key candidate k_9 is associated with the pair of classes (Woman $\sqcap_{class}$ Scientist, FemScientist), i.e., k_9 relates an instance of the class conjunction Woman $\sqcap_{class}$ Scientist with an instance of the class FemScientist. Moreover, different link key candidates may have the same sets of pairs of properties Eq and In, but they are associated with different pairs of class expressions, as this is the case for k_{1a} and k_{1b}. Indeed, k_{1a} and k_{1b} have the same sets of properties pairs, namely {(desig,title)}, {(desig,title)}, while they are associated respectively with (Book,Dictionary $\sqcup_{class}$ Novel) and (Book,Novel).

The properties appearing in such link key candidates describe subjects belonging to different classes. For example, the property title in k_{1a} and k_{1b} describes instances of the classes Dictionary or Novel. Furthermore, it should be noticed that one pair of classes can be associated with more than one link key candidate. For example, the pair of classes (Book,Novel) (in orange in Figure 4) is associated with the four link key candidates k_3, k_4, k_5, and k_6. In particular, this means that there are four possible choices to select a link key candidate for the pair of classes (Book,Novel).

Now let us have a more precise look at the link sets generated by link key candidates. For example, the link set $L(k_{11})$ = {(a_1,b_1),(a_2,b_2),(a_2,b_1)} is generated by k_{11} and $L(k_{12})$ = {(a_1,b_1),(a_2,b_2),(a_1,b_2)} is generated by k_{12}. An identity link states that two subjects represent the same entity and materialized by the owl:sameAs relation. The latter is an equivalence relation, i.e., reflexive, symmetric, and transitive. Relying on the transitivity of owl:sameAs and considering link sets as partitions, it comes that k_{11} and k_{12} are generating the same equivalence class of links [18]. Then it can be concluded that the four subjects a_1, a_2, b_1, and b_2, represent the same entity.

Therefore, when link sets are considered as partitions induced by owl:sameAs, the two link key candidates k_{11} and k_{12} are carrying the same information about identity links. In other words, this involves a kind of "redundancy", i.e., it is

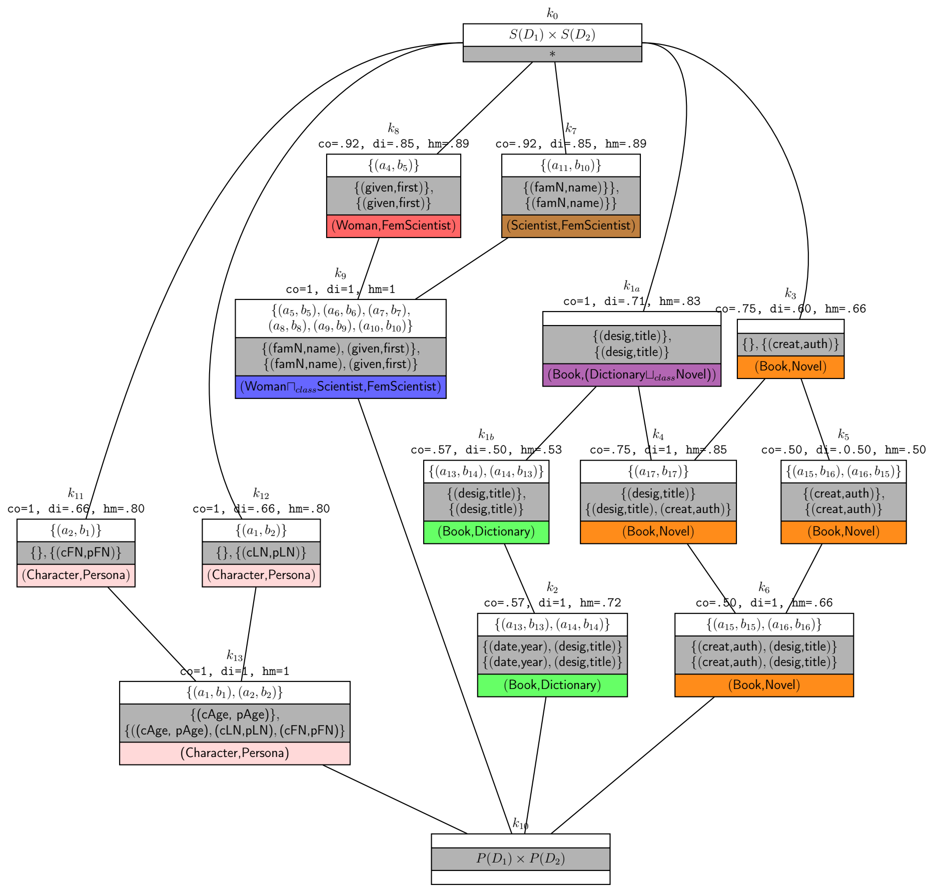
k_0
$S(D_1) \times S(D_2)$
*
k_8
co=.92, di=.85, hm=.89
$\{(a_4, b_5)\}$
{(given,first)}, {(given,first)}
(Woman,FemScientist)
k_7
co=.92, di=.85, hm=.89
$\{(a_{11}, b_{10})\}$
{(famN,name)}, {(famN,name)}
(Scientist,FemScientist)
k_9
co=1, di=1, hm=1
$\{(a_5, b_5), (a_6, b_6), (a_7, b_7), (a_8, b_8), (a_9, b_9), (a_{10}, b_{10})\}$
{(famN,name), (given,first)}, {(famN,name), (given,first)}
(Woman$\sqcap_{class}$Scientist,FemScientist)
k_{1a}
co=1, di=.71, hm=.83
{(desig,title)}, {(desig,title)}
(Book,(Dictionary$\sqcup_{class}$Novel))
k_3
co=.75, di=.60, hm=.66
{}, {(creat,auth)}
(Book,Novel)
k_{1b}
co=.57, di=.50, hm=.53
$\{(a_{13}, b_{14}), (a_{14}, b_{13})\}$
{(desig,title)}, {(desig,title)}
(Book,Dictionary)
k_4
co=.75, di=1, hm=.85
$\{(a_{17}, b_{17})\}$
{(desig,title)} {(desig,title), (creat,auth)}
(Book,Novel)
k_5
co=.50, di=.0.50, hm=.50
$\{(a_{15}, b_{16}), (a_{16}, b_{15})\}$
{(creat,auth)}, {(creat,auth)}
(Book,Novel)
k_{11}
co=1, di=.66, hm=.80
$\{(a_2, b_1)\}$
{}, {(cFN,pFN)}
(Character,Persona)
k_{12}
co=1, di=.66, hm=.80
$\{(a_1, b_2)\}$
{}, {(cLN,pLN)}
(Character,Persona)
k_2
co=.57, di=1, hm=.72
$\{(a_{13}, b_{13}), (a_{14}, b_{14})\}$
{(date,year), (desig,title)} {(date,year), (desig,title)}
(Book,Dictionary)
k_6
co=.50, di=1, hm=.66
$\{(a_{15}, b_{15}), (a_{16}, b_{16})\}$
{(creat,auth), (desig,title)} {(creat,auth), (desig,title)}
(Book,Novel)
k_{13}
co=1, di=1, hm=1
$\{(a_1, b_1), (a_2, b_2)\}$
{(cAge, pAge)}, {((cAge, pAge), (cLN,pLN), (cFN,pFN)}
(Character,Persona)
k_{10}
$P(D_1) \times P(D_2)$

Figure 4: The pattern concept lattice induced from the LK-pattern structure in Table 1.

no longer true that every link key candidate provides information that cannot be retrieved from the rest of the link key candidates. Accordingly, the topic of the next Section is to present how to define, to detect, and to deal with such redundancy.

5. Detecting Redundant Link Keys with Partition Pattern Structures

5.1. *Detecting the Redundancy of Link Keys based on Partition Pattern Structures*

In [19], authors are using "partition pattern structures" for discovering functional dependencies (FDs), which correspond to attribute implications or to pairs of attributes having the same behavior w.r.t. an object partition. Here we are facing the same kind of problem and thus we reuse the same approach, replacing "implication" by "link key candidate", knowing that an implication within a concept lattice is related with a closed set as link key candidates are.

Below, we show how the detection of redundancy among link key candidates is based on the use of partition pattern structures, where partitions of subjects are composed of equivalent subsets of subjects in generated link sets. Firstly, we present a motivating example explaining the process of redundancy detection. In Figure 4, the link key candidate k_{1b} generates the following link set:

$\mathsf{L}(\mathsf{k}_{1b}) = \{(\mathsf{a}_{13},\mathsf{b}_{13}),(\mathsf{a}_{14},\mathsf{b}_{14}),(\mathsf{a}_{13},\mathsf{b}_{14}),(\mathsf{a}_{14},\mathsf{b}_{13})\}$.

Based on the symmetry and transitivity of the owl:sameAs relation, it is possible to compute the equivalence class of subjects lying in the $\mathsf{L}(\mathsf{k}_{1b})$ link set. Then, the link $(\mathsf{a}_{13},\mathsf{a}_{14})$ can be inferred from the pairs $(\mathsf{a}_{13},\mathsf{b}_{13})$ and $(\mathsf{b}_{13},\mathsf{a}_{14})$ (using symmetry for the latter). In particular, these two pairs can be rewritten as the RDF triples $(\mathsf{a}_{13}$,owl:sameAs,$\mathsf{b}_{13})$ and $(\mathsf{b}_{13}$,owl:sameAs,$\mathsf{a}_{14})$. Computing the transitive closure of the owl:sameAs relation in $\mathsf{L}(\mathsf{k}_{1b})$ yields the partition $\{\mathsf{a}_{13},\mathsf{a}_{14},\mathsf{b}_{13},\mathsf{b}_{14}\}$. This partition can be regarded as an equivalence class or as a "complete graph" –all vertices are connected to each others. Based on that, we have the following definition:

Definition 9 (Partition of subjects induced by a link key)**.** Let D_1 and D_2 be two RDF datasets and k a link key candidate. The *partition of subjects induced by k* is defined as the quotient set: $Part(k) = L(k)/$owl:sameAs.

Before making precise the "partition pattern structure" supporting the discovery of such quotient sets, we recall that the meet of two partitions $\sqcap_{part}$ is defined as the coarsest common refinement of the two partitions [19]. For example:

$\{\{\mathsf{a}_{15},\mathsf{a}_{16},\mathsf{b}_{15},\mathsf{b}_{16}\}\} \sqcap_{part} \{\{\mathsf{a}_{15},\mathsf{b}_{15}\},\{\mathsf{a}_{16},\mathsf{b}_{16}\},\{\mathsf{a}_{17},\mathsf{b}_{17}\}\} = \{\{\mathsf{a}_{15},\mathsf{b}_{15}\},\{\mathsf{a}_{16},\mathsf{b}_{16}\}\}$.

Given two RDF datasets D_1 and D_1, the set of instances $I(D_1) \cup I(D_1)$, the related set of partitions equipped with the similarity operator $\sqcap_{part}$, we can build the following "partition pattern structure":

Definition 10 (LK-partition pattern structure)**.** Let D_1 and D_2 be two RDF datasets. The *LK-partition pattern structure* over D_1 and D_2 is defined as the triple $(LKC, (Part(S(D_1) \cup S(D_2)), \sqcap_{part}), \delta)$ such that:

– LKC is the set of link key candidates over D_1 and D_2 computed thanks to the LK-pattern structure introduced in Section 4,

– $Part(S(D_1) \cup S(D_2))$ is the set of partitions of the set of all subjects in D_1 and D_2, while $\sqcap_{part}$ is the "meet" of two partitions,

– δ maps a link key candidate k to the partition $Part(k) = L(k)/$owl:sameAs, where $L(k)$ is the link set generated by k.

For example, the LK-partition pattern structure corresponding to the RDF datasets displayed in Figure 1 is given in Table 2. The objects are the link key candidates k_{1a}, k_{1b}, $\mathsf{k}_2, \ldots, \mathsf{k}_9$, k_{11}, k_{12}, k_{13}, which are taken from the LK-pattern structure given in Table 1, while the descriptions are the related partitions of subjects.

PPS objects (k_i)	Descriptions ($\delta(k_i) = Part(k_i)$)
k_{1a}	$\{\{a_{13}, a_{14}, b_{13}, b_{14}\}, \{a_{15}, b_{15}\}, \{a_{16}, b_{16}\}, \{a_{17}, b_{17}\}\}$
k_{1b}	$\{\{a_{13}, a_{14}, b_{13}, b_{14}\}\}$
k_2	$\{\{a_{13}, b_{13}\}, \{a_{14}, b_{14}\}\}$
k_3	$\{\{a_{15}, a_{16}, b_{15}, b_{16}\}, \{a_{17}, b_{17}\}\}$
k_4	$\{\{a_{15}, b_{15}\}, \{a_{16}, b_{16}\}, \{a_{17}, b_{17}\}\}$
k_5	$\{\{a_{15}, a_{16}, b_{15}, b_{16}\}\}$
k_6	$\{\{a_{15}, b_{15}\}, \{a_{16}, b_{16}\}\}$
k_7	$\{\{a_5, b_5\}, \{a_6, b_6\}, \{a_7, b_7\}, \{a_8, b_8\}, \{a_9, b_9\}, \{a_{10}, a_{11}, b_{10}\}\}$
k_8	$\{\{a_4, a_5, b_5\}, \{a_6, b_6\}, \{a_7, b_7\}, \{a_8, b_8\}, \{a_9, b_9\}, \{a_{10}, b_{10}\}\}$
k_9	$\{\{a_5, b_5\}, \{a_6, b_6\}, \{a_7, b_7\}, \{a_8, b_8\}, \{a_9, b_9\}, \{a_{10}, b_{10}\}\}$
k_{11}	$\{\{a_1, a_2, b_1, b_2\}\}$
k_{12}	$\{\{a_1, a_2, b_1, b_2\}\}$
k_{13}	$\{\{a_1, b_1\}, \{a_2, b_2\}\}$

Table 2: The "partition pattern structure" corresponding to the pattern structure given in Table 1.

Pattern concepts from this LK-partition pattern structure are composed of a set of link keys (the extent) and a partition of the set of instances (the intent) that is the coarsest refinement of the partitions induced by the link keys in the extent. In order to detect redundant link keys, we want to find sets of link keys that induce the same partition of instances, i.e. link keys k_i, k_j such that $Part(k_i) = Part(k_j)$. As such, we do not need all the concepts of this LK-partition pattern structure but only the concepts in which the intent is a partition of subjects induced by a single link key, i.e. concepts of the form $(k_i^{\diamond\diamond}, k_i^{\diamond})$. Those concepts are, by definition, the object concepts introducers (about the OC-poset, see Section 3 and [22]). If the extent of these concepts contains more than one element, it contains redundant link keys.

Figure 5 shows the OC-poset of the running example completed into a lattice (for the sake of readability). We can observe that the link key candidates k_{11} and k_{12} are merged into the same concept, at the bottom left. This concept can be interpreted in the following way: k_{11} and k_{12} are inducing the same partition and they should be considered as equivalent. By contrast, the extents of the other concepts contain only a single element as these link key candidates are not redundant.

5.2. From Partition Equality to Partition Similarity

Partitions of subjects induced by link keys can be different but still close. Then, instead of comparing partitions thanks to equality, one can use an adapted similarity measure. Given two partitions, such a similarity measure can be defined as the ratio of the common equivalence classes among all equivalence classes composing both partitions.

Definition 11 (Similarity between partitions)**.** Given two partitions P_1 and P_2, the similarity measure between P_1 and P_2 can be defined as:

$$\sigma(P_1, P_2) = \frac{|P_1 \cap P_2|}{|P_1 \cup P_2|}$$

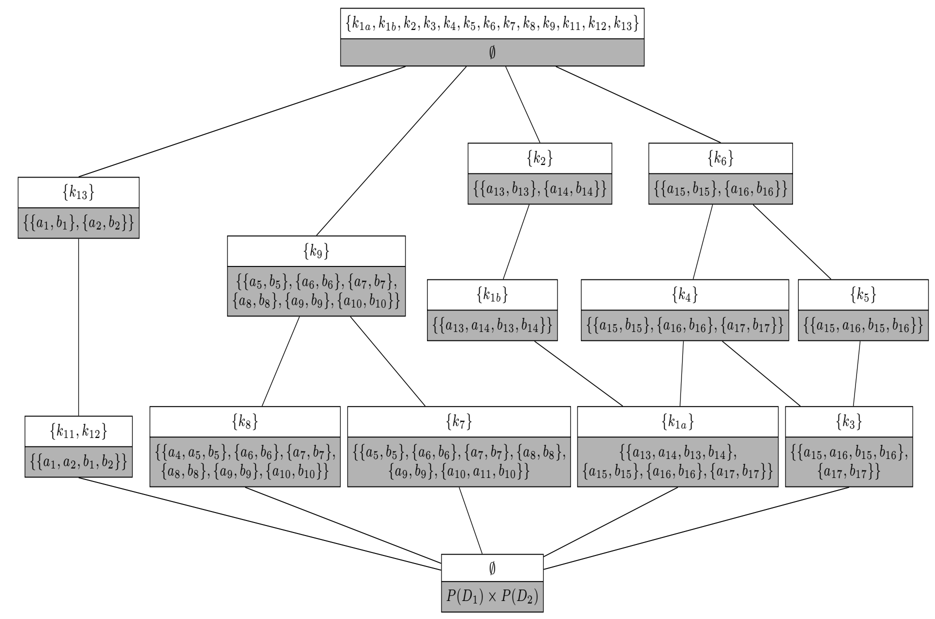
$\{k_{1a}, k_{1b}, k_2, k_3, k_4, k_5, k_6, k_7, k_8, k_9, k_{11}, k_{12}, k_{13}\}$
$\emptyset$
$\{k_{13}\}$
$\{\{a_1, b_1\}, \{a_2, b_2\}\}$
$\{k_9\}$
$\{\{a_5, b_5\}, \{a_6, b_6\}, \{a_7, b_7\}, \{a_8, b_8\}, \{a_9, b_9\}, \{a_{10}, b_{10}\}\}$
$\{k_2\}$
$\{\{a_{13}, b_{13}\}, \{a_{14}, b_{14}\}\}$
$\{k_6\}$
$\{\{a_{15}, b_{15}\}, \{a_{16}, b_{16}\}\}$
$\{k_{1b}\}$
$\{\{a_{13}, a_{14}, b_{13}, b_{14}\}\}$
$\{k_4\}$
$\{\{a_{15}, b_{15}\}, \{a_{16}, b_{16}\}, \{a_{17}, b_{17}\}\}$
$\{k_5\}$
$\{\{a_{15}, a_{16}, b_{15}, b_{16}\}\}$
$\{k_{11}, k_{12}\}$
$\{\{a_1, a_2, b_1, b_2\}\}$
$\{k_8\}$
$\{\{a_4, a_5, b_5\}, \{a_6, b_6\}, \{a_7, b_7\}, \{a_8, b_8\}, \{a_9, b_9\}, \{a_{10}, b_{10}\}\}$
$\{k_7\}$
$\{\{a_5, b_5\}, \{a_6, b_6\}, \{a_7, b_7\}, \{a_8, b_8\}, \{a_9, b_9\}, \{a_{10}, a_{11}, b_{10}\}\}$
$\{k_{1a}\}$
$\{\{a_{13}, a_{14}, b_{13}, b_{14}\}, \{a_{15}, b_{15}\}, \{a_{16}, b_{16}\}, \{a_{17}, b_{17}\}\}$
$\{k_3\}$
$\{\{a_{15}, a_{16}, b_{15}, b_{16}\}, \{a_{17}, b_{17}\}\}$
$\emptyset$
$P(D_1) \times P(D_2)$

Figure 5: The OC-poset of the LK-partition pattern structure given in Table 2.

For example, in Table 2, the similarity between $\mathsf{Part}(\mathsf{k}_3)$ and $\mathsf{Part}(\mathsf{k}_4)$ is 1/4 because $\mathsf{Part}(\mathsf{k}_3)$ and $\mathsf{Part}(\mathsf{k}_4)$ only share one equivalence class, namely $\{\mathsf{a}_{17},\mathsf{b}_{17}\}$. By contrast, the similarity between partitions $\mathsf{Part}(\mathsf{k}_5)$ and $\mathsf{Part}(\mathsf{k}_6)$ is 0 because no equivalence class is shared, even if there exists a subsumption relation between k_5 and k_6 (see Figure 5).

In data clustering, the Rand index is usually used for comparing partitions [23]. Since this index is defined over pairs of elements, it may return a more precise estimation of similarity. However, it requires that the two compared partitions are defined over the same set of objects, which is not the case here.

Here after, in Section 6, we present experiments about the redundancy of link keys in real-world datasets. Moreover, we also discuss the contrast existing between the use of equality and the use of similarity for comparing partitions. Actually, redundancy which is not really significant when based on equality becomes much more significant when based on similarity.

6. Experiments

There are two main objectives in these experiments. The first one is to show how the discovery of link keys based on pattern structure extends the related process based on FCA as proposed in [12]. The second one is to investigate in real-world datasets how significant is redundancy among link keys, when partition pattern structures are used to detect such redundancy.

Task	Dataset	classes	#inst.	#prop.	#triples	|LK-PS|
Restaurants	Restaurant1	r1:Restaurant, r1:Address, r1:City	339	6	1130	36
	Restaurant2	r2:Restaurant, r2:Address, r2:Category	2256	6	7520	
Person1	Person11	p11:Person, p11:Address, p11:Suburb, p11:State	2000	13	9000	2212
	Person12	p12:Person, p12:Address	1000	12	7000	
Pr	Pr1	Classes of Restaurant1 and Person11	2339	19	10130	2260
	Pr2	Classes of Restaurant2 and Person12	3256	18	14520	
Actor	DB Actor	db:Actor	5807	16	94606	2005
	Yago Actor	yago:Actor	108415	16	1029580	

Table 3: The description of the datasets used in the experiments. The columns #inst., #prop., and #triples, respectively denote the number of instances, of properties, and of triples in each dataset. The number of link key candidates generated in the pattern structure is given in the column |LK-PS|.

6.1. The Datasets and Their Contents

In these experiments, the so-called "tasks" described in Table 3 are taken from OAEI[1] 2010. A task consists in discovering links between instances of the two corresponding datasets given in the column "Dataset". For every task, a set of reference owl:sameAs links is provided (gold standard). Accordingly, reference links between classes r1:Restaurant and r2:Restaurant are provided in the task Restaurants, and between classes p11:Person and p12:Person as well. By contrast, as they were not provided, we built the reference links between r1:Address and r2:Address, and between classes p11:Address and p12:Address The task Pr, built on purpose, is based on a combination of Restaurant1 and Person11 from one side, and a combination of Restaurant2 and Person12 from the other side. Moreover, we considered the task Actor introduced in [24] which consists in linking the class db:Actor from DBpedia and the class yago:Actor from Yago. A set of reference links between these two classes is provided.

Regarding the provenance of these datasets, Restaurant1 and Person1 are taken from the well-known "Ontology Alignment Evaluation Initiative (OAEI)" which includes a collection of benchmarks designed to assess and to compare the performance of ontology alignment systems, including data interlinking systems. Experiments were conducted on additional datasets taken from OAEI, namely Doremus (OAEI 2016) and SPIMBench (OAEI 2018), and similar results were observed.

Finally, experiments were also conducted on the Actor dataset, which was extracted from DBpedia and Yago. This dataset was selected because of its particular use in experiments about data interlinking based on keys and conditional keys [24]. Additional results about these last experiments can be found in [18].

[1]`http://oaei.ontologymatching.org/2010/im/index.html`

6.2. A Pattern Structure for Link Key Discovery

As pointed out above, one objective in these experiments is to understand to which extent link key discovery based on pattern structure extends the FCA-based approach proposed in [12]. Thus link key candidates were generated using both methods and then ranked thanks to hm, i.e., the harmonic mean of coverage and discriminability. The results returned by the FCA-based process can be interpreted as follows:

- The top link key candidate for task Restaurants, say $\mathtt{k}_{\mathtt{R}}^{\mathtt{1st}}$, generates links between classes r1:Restaurant and r2:Restaurant, while $\mathtt{k}_{\mathtt{R}}^{\mathtt{1st}}$ does not generate any link between classes r1:Address and r2:Address. The link key generating links between r1:Address and r2:Address is ranked second.

- The top link key candidate for task Person1, say $\mathtt{k}_{\mathtt{P1}}^{\mathtt{1st}}$, generates links between classes p11:Person and p12:Person, while $\mathtt{k}_{\mathtt{P1}}^{\mathtt{1st}}$ does not generate any link between classes p11:Address and p12:Address. Moreover, the link key candidates ranked from 1 to 8 do not generate any link between classes p11:Address and p12:Address.

- The top link key candidate for task Pr, $\mathtt{k}_{\mathtt{Pr}}^{\mathtt{1st}}$, generates links between classes p11:Person and p12:Person only. The first link key candidate generating links between classes p11:Address and p12:Address is ranked 8^{th}.

  The first link key candidate generating links between classes r1:Restaurant and r2:Restaurant is ranked at position 1976. Moreover, the first link key candidate generating links between classes r1:Address and r2:Address is ranked at position 2049.

- Finally, the top link key candidate for task Actor generates links between db:Actor and yago:Actor, which are the most general classes in this task. Then the interest of such link key candidates is rather low.

The FCA-based method for discovering link key candidates returns candidates without specifying the pairs of classes, such as ({}, {(r1:name,r2:name)}, ()), which is the top link key candidate in the task Restaurants. This link key candidate can be interpreted as follows: if an instance of Restaurant1 and an instance of Restaurant2 share at least a value for the property name, then these two instances denote the same entity. Even a non expert person may conclude that the validity of such a link key is not very high, since holding the same name does not mean at all lying at the same place.

By contrast, the use of an adapted pattern structure as discussed in this paper allows one to discover link key candidates with related pairs of classes. Then it is straightforward to select link key candidates w.r.t. a target pair of classes. For example, in the task Restaurants, it is possible to select if required the top link key candidate defined over classes r1:Restaurant and r2:Restaurant, namely ({},{(r1:name,r2:name)},(r1:Restaurant,r2:Restaurant)). Then, the first link key candidate defined over classes r1:Address and r2:Address can be selected in

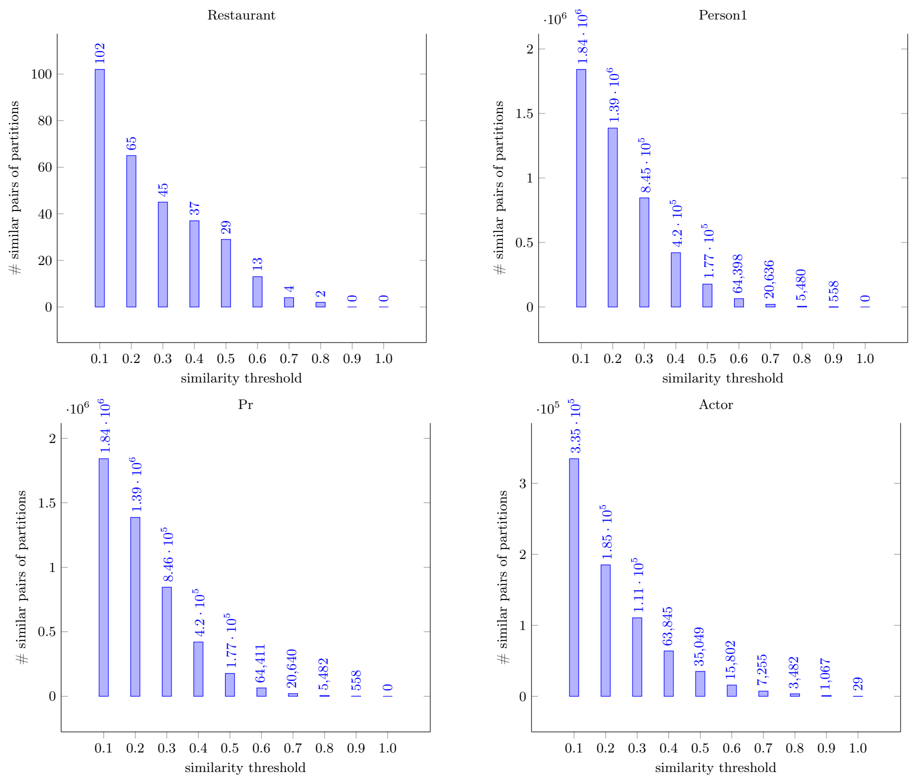

Figure 6: Variation of the number of similar pairs of partitions w.r.t. the similarity threshold. In this experiment, we do not count pairs of the same partition (P, P) (reflexivity) and we count pairs (P_1, P_2) and (P_2, P_1) as only one pair (symmetry).

continuation, namely ({},{(r1:street,r2:street)},(r1:Address,r2:Address)). This process ensures a progressive selection process of link key candidates, where any class pair can be considered, excluding inconsistent or not relevant pairs.

Let us consider now the link key candidate ({},{(r1:name,p12:given_name)}, (r1:Restaurant,p12:Person)) that is returned in task Pr. Actually, this link key candidate instantiates an inconsistent pair of classes, since a restaurant and a person cannot materialize the same entity. As such this candidate should be excluded by a domain expert. Accordingly, the paper [25] follows this idea and proposes the LKSA algorithm aimed at the selection of link key candidate guided by the associated pairs of classes. This enforces the fact that link key candidates with associated pairs of classes are significant and bring an added value in link key management.

6.3. The Redundancy of Link key candidates

The next experiments are aimed at determining how important is redundancy of link key candidates in real-world datasets. Firstly, the set of link key candidates, named "LK-PS", is generated thanks to the LK-pattern structure. Then the redundancy of candidates is based on the equality of partitions in the LK-partition pattern structure, i.e., two link key candidates are declared as redundant when they induce the same link partition. Then, the set of link key candidates lying in the OC-poset, related to the LK-partition pattern structure and named "LK-PPS", can be generated. The results can be read as follows. In the tasks Restaurants, Person1, and Pr, it comes that |LK-PS| = |LK-PPS|. Thus no redundancy among link keys can be observed in these real-world datasets. Besides that, in task Actor, there are 29 pairs of partitions which are equal, showing that some redundant candidates are existing in these datasets. More precisely, |LK-PPS| is lower than |K-PS| by 1%.

Hence, it can be concluded that redundancy of link key candidates based on equality is not really significant in the real-world datasets considered in these experiments. This also shows that equality of partitions is probably a too strong constraint and should be relaxed. Following this line, the next experiments are no longer based on equality of partitions but instead on similarity of partitions as introduced in Definition 11. The similarity measure is evaluated for all pairs of candidates contained in LK-PS. Figure 6 shows the cumulative distributions for each task.

The results can be read as follows. The link key candidates generate similar partitions when the similarity threshold is equal or below 0.8 in task Restaurants, and when the similarity threshold is equal or below 0.9 in tasks Person 1 and Pr.

In the task Actor, it can be observed that 29 pairs of link key candidates are generating the same partitions when the similarity threshold is $1,0$. By contrast, this number exponentially increases when the threshold is set to 0.9, as the number of similar pairs of partitions jumps from 29 to 1067! More generally, such a break can be verified in all the datasets with threshold values varying from 0.5 to 0.9, except in Restaurants where the number of pairs is smaller than in the other datasets.

Contrasting the results of the experiments based on equality of partitions, the experiments based on similarity show that a "redundancy" effectively exists in real-world datasets, even if it is rather weak. Then redundancy based on similarity of partitions can be considered as a valuable criterion for merging link key candidates, especially in large datasets. Indeed, the number of link key candidates can increase very quickly, and then the resulting pattern concept lattices become much harder to visualize, to handle, and to interpret. Merging redundant link keys based on similarity of partitions is a way to minimize the total number of link keys and to build kinds of "classes" of link keys. Then such classes of link keys can be proposed to domain experts for evaluation and interpretation. More should be done in this way and this opens new lines of research for future work.

7. Related Work

Data interlinking is a difficult task is studied since a long time. Systems such as "SILK" [4] and "LIMES" [3] rely on link specifications to discover links, and in particular identity links, between two RDF datasets. Link specifications define conditions that subjects should meet in order to be linked. These specifications are complex, as they are composed of many elements, (i) properties to be compared, (ii) transformations to be applied, (iii) similarity measures to be used for comparing pairs, (iv) aggregation functions for combining several similarity values, and finally (v) similarity thresholds. To optimize link generation in terms of runtime efficiency and scaling, SILK relies on blocking strategies. By contrast, LIMES takes advantage of the triangular inequality in the Euclidean space for reducing the number of comparisons.

Moreover, in SILK and LIMES, while users may manually define link specifications, algorithms such as "RAVEN" [26], "EAGLE" [6], and "LIGON" [27], are used to automatically discover links between datasets. One main common characteristic shared by these algorithms is that they are based on machine learning techniques, and thus they require a set of reference links. Furthermore, they do not take into account the semantics of ontologies possibly attached to the considered datasets, which describe elements in the RDF triples and provide classes.

As underlined above, link specifications require the definition of properties and classes to be compared, as well as determining the adapted similarity measures and associated thresholds. To a certain extent, link keys can be considered as a particular kind of link specifications. In this way, link key discovery techniques can be valuable assets and complements for supporting approaches based on link specifications, as they automatically provide properties and classes to be compared without requiring reference links. Such properties and classes provide a support for link specification along with similarity measures and thresholds.

There are some other approaches in link discovery relying on logical axioms from which are deriving the researched links. As they are based on a logical interpretation, logical reasoners or rule interpreters can take advantage of logical axioms to infer such new facts. These approaches are more efficient when the descriptions of the same entity are likely to be identical. In particular, key-based approaches are falling in this category. A key is a set of properties whose values unequivocally define instances of the associated class within a dataset. Although keys are not directly intended to be used in data interlinking, they can easily be used for solving this task, as shown in the following example. Let us consider two datasets, say D_1 and D_2, and the following key related to D_1 and D_2, namely "{firstName,lastName}, (Person)". Here firstName and lastName are properties, and Person is a class of individuals related to D_1 and D_2. The key can be interpreted as follows: when an instance say p_1 of Person in D_1 and an instance say p_2 of the same class Person in D_2 have the same firstName and the same lastName, then p_1 and p_2 represent the same entity, and an identity link can be created for relating p_1 and p_2. As one can see, based on the above description, keys are very close to link keys.

Keys, as this is also the case for link keys, are not necessarily provided in datasets, and various approaches have been proposed to automatically discovering such keys. For example, the "KD2R" algorithm [28] discovers keys that are valid in multiple RDF datasets. However, when the datasets contain duplicates or erroneous data, no keys can be discovered using this approach. Furthermore, the KD2R algorithm is hardly scalable when dealing with large datasets. Thus a new algorithm called "SAKey" was introduced in [29] to improve applicability and scalability of the precedent key-based approach. SAKey discovers keys with exceptions called *n-almost keys*. A set of properties is an n-almost key if at most n instances share the same values for this set of properties. A later improvement was brought in the "VICKEY" algorithm [24] for dealing with the absence of keys in some datasets. VICKEY is able to discover the so-called "conditional keys" that are valid only in a part of the whole dataset where specific conditions made precise within the conditional key are met. When they are running, KD2R, SAKey, and VICKEY, are following the same main steps. Firstly, they compute a set of maximal non-keys, which are based on combinations of properties sharing the same values for at least two instances. Then the keys are computed from the set of non-keys. Moreover, these algorithms make use of various efficient techniques to reduce the search space of potential keys.

A closer look on the way keys are used in data interlinking shows that the two datasets considered for interlinking should rely on the same ontology or an ontology alignment has to be provided. This can be seen as a limitation that can be overcome by using link keys. Then, as this is the case for keys, link keys can be declared as logical axioms. By contrast, unlike keys, link keys can be used in data interlinking in scenarios when different ontologies are involved or when no ontology alignment is provided. Finally, an experimental comparison between the present approach based on link keys, and approaches based on keys and conditional keys, in terms of recall, precision, and F-measure, is detailed in [18].

In the FCA community, a recent work dealing with data interlinking was proposed in [30]. More precisely, the Graph-FCA system [31] is aimed at extending the FCA setting to deal with multi-relational or graph data. Graph-FCA can be used to discover identity links between instances as well as alignments between properties and classes. Graph-FCA and the approach presented in this paper share some commonalities, such as discovering identity links between instances in different datasets. However, the objectives of the two systems are quite different, dealing with graph data for the former and dealing with link key discovery for the latter. Nevertheless, a careful comparison of both approaches remains to be carried out.

8. Conclusion

In this paper, we have made precise an original research work about data interlinking based on FCA and Pattern Structures. The main objective is to discover the so-called link key candidates and the related identity link sets across two RDF datasets in the web of data. Actually, link keys are syntactic constructions based on pairs of properties and of classes that can be considered as rules allowing to infer identity links between two subjects in RDF datasets. FCA and pattern structures are well adapted to support this discovery process. FCA is used to discover link key candidates while pattern structures allow one to associate pair of classes with a link key candidate. Moreover, when the owl:sameAs relation is regarded as an equivalence relation, two link key candidates can be considered as redundant when they generate the same link sets. Then partition pattern structures allow one to study the potential redundancy of link key candidates. It appears that FCA and pattern structures are really valuable formalisms in link key discovery, and this paper demonstrates it.

Future work can be carried out in several directions. Firstly, there is a need for improving the formalization of link key discovery within the FCA and the pattern structure formalisms. In particular, more complete investigations about the relations between equality-based and similarity-based partitions, and as well the significance of redundancy remain to be done. In parallel, more experiments should be performed to support and demonstrate these theoretical aspects. Finally, another direction is to study the relations existing with link discovery in Knowledge Graphs and more generally in the web of data [32, 33], especially link prediction [34] and entity summarization [35].

Acknowledgments

This work has been partially supported by the ANR project Elker (ANR-17-CE23-0007-01).

References

[1] M. Nentwig, M. Hartung, A.-C. Ngonga Ngomo, E. Rahm, A survey of current Link Discovery frameworks, Semantic Web Journal 8 (3) (2017) 419–436.

[2] A. Hogan, The Web of Data, Springer, 2020.

[3] A. N. Ngomo, S. Auer, LIMES – A Time-Efficient Approach for Large-Scale Link Discovery on the Web of Data, in: T. Walsh (Ed.), Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI), IJCAI/AAAI, 2011, pp. 2312–2317.

[4] J. Volz, C. Bizer, M. Gaedke, G. Kobilarov, Silk – A Link Discovery Framework for the Web of Data, in: C. Bizer, T. Heath, T. Berners-Lee, K. Idehen (Eds.), Proceedings of the WWW2009 Workshop on Linked Data on the Web (LDOW), CEUR Workshop Proceedings 538, 2009.

[5] R. Isele, C. Bizer, Active learning of expressive linkage rules using genetic programming, Journal of web semantics 23 (2013) 2–15.

[6] A. N. Ngomo, K. Lyko, EAGLE: Efficient Active Learning of Link Specifications Using Genetic Programming, in: E. Simperl, P. Cimiano, A. Polleres, Ó. Corcho, V. Presutti (Eds.), Proceedings of the 9th Extended Semantic Web Conference (ESWC), Lecture Notes in Computer Science 7295, Springer, 2012, pp. 149–163.

[7] A. N. Ngomo, K. Lyko, Unsupervised learning of link specifications: deterministic vs. non-deterministic, in: P. Shvaiko, J. Euzenat, K. Srinivas, M. Mao, E. Jiménez-Ruiz (Eds.), Proceedings of the 8th International Workshop on Ontology Matching at ISWC, CEUR Workshop Proceedings 1111, 2013, pp. 25–36.

[8] F. Saïs, N. Pernelle, M.-C. Rousset, L2R: A Logical Method for Reference Reconciliation, in: Proc. 22nd National Conference on Artificial Intelligence (AAAI), Vancouver (CA), AAAI Press, 2007, pp. 329–334.

[9] M. Al-Bakri, M. Atencia, S. Lalande, M.-C. Rousset, Inferring same-as facts from Linked Data: an iterative import-by-query approach, in: Proceedings of the 29th AAAI Conference on Artificial Intelligence, Austin (TX US), AAAI Press, 2015, pp. 9–15.

[10] M. Al-Bakri, M. Atencia, J. David, S. Lalande, M.-C. Rousset, Uncertainty-sensitive reasoning for inferring sameAs facts in linked data, in: Proceedings of the 22nd European Conference on Artificial intelligence (ECAI), Den Haague (NL), 2016, pp. 698–706.

[11] M. Atencia, J. David, J. Euzenat, Data interlinking through robust linkkey extraction, in: Proceedings of ECAI, 2014, pp. 15–20.

[12] M. Atencia, J. David, J. Euzenat, A. Napoli, J. Vizzini, Link key candidate extraction with relational concept analysis, Discrete Applied Mathematics 273 (2020) 2–20.

[13] N. Abbas, J. David, A. Napoli, Discovery of Link Keys in RDF Data Based on Pattern Structures: Preliminary Steps, in: F. J. Valverde-Albacete, M. Trnecka (Eds.), Proceedings of the International Conference on Concept Lattices and Their Applications (CLA), CEUR Workshop Proceedings 2668, 2020, pp. 235–246.

[14] M. Atencia, J. David, J. Euzenat, What can FCA do for database linkkey extraction?, in: S. O. Kuznetsov, A. Napoli, S. Rudolph (Eds.), Proceedings of FCA4AI Workshop, CEUR Proceedings Vol-1257, 2014, pp. 85–92.

[15] B. Ganter, R. Wille, Formal Concept Analysis: Mathematical Foundations, Springer, 1999.

[16] B. Ganter, S. O. Kuznetsov, Pattern Structures and Their Projections, in: H. S. Delugach, G. Stumme (Eds.), Proceedings of the International Conference on Conceptual Structures (ICCS), Lecture Notes in Computer Science 2120, Springer, 2001, pp. 129–142.

[17] M. Kaytoue, S. O. Kuznetsov, A. Napoli, S. Duplessis, Mining Gene Expression Data with Pattern Structures in Formal Concept Analysis, Information Sciences 181 (10) (2011) 1989–2001.

[18] N. Abbas, A. Bazin, J. David, A. Napoli, A Study of the Discovery and Redundancy of Link Keys Between Two RDF Datasets Based on Partition Pattern Structures, in: P. C. Ortega, O. Krídlo (Eds.), Proceedings of the International Conference on Concept Lattices and Their Applications (CLA), CEUR Workshop Proceedings 3308, 2022, pp. 175–189.

[19] J. Baixeries, M. Kaytoue, A. Napoli, Characterizing functional dependencies in formal concept analysis with pattern structures, Annals of Mathematics and Artificial Intelligence 72 (2014) 129–149.

[20] F. Baader, D. Calvanese, D. McGuinness, P. Patel-Schneider, D. Nardi, et al., The Description Logic Handbook: Theory, Implementation and Applications, Cambridge University Press, 2003.

[21] M. Atencia, J. David, J. Euzenat, On the relation between keys and link keys for data interlinking, Semantic Web Journal 12 (4) (2021) 547–567.

[22] A. Berry, A. Gutierrez, M. Huchard, A. Napoli, A. Sigayret, Hermes: a simple and efficient algorithm for building the AOC-poset of a binary relation, Annals of Mathematics and Artificial Intelligence 72 (2014) 45–71.

[23] W. M. Rand, Objective criteria for the evaluation of clustering methods, Journal of the American Statistical Association 66 (336) (1971) 846–850.

[24] D. Symeonidou, L. Galárraga, N. Pernelle, F. Saïs, F. M. Suchanek, VICKEY: Mining Conditional Keys on Knowledge Bases, in: Proceedings of 16th International Semantic Web Conference (ISWC), LNCS 10587, Springer, 2017, pp. 661–677.

[25] N. Abbas, J. David, A. Napoli, LKSA : un algorithme de sélection de clés de liage dans des données RDF guidée par des paires de classes, in: J. Azé, V. Lemaire (Eds.), Extraction et Gestion des Connaissances (EGC), Vol. E-37 of RNTI, Éditions RNTI, 2021, pp. 205–216.

[26] A. N. Ngomo, J. Lehmann, S. Auer, K. Höffner, RAVEN – Active Learning of Link Specifications, in: P. Shvaiko, J. Euzenat, T. Heath, C. Quix, M. Mao, I. F. Cruz (Eds.), Proceedings of the 6th International Workshop on Ontology Matching, CEUR Workshop Proceedings 814, 2011.

[27] M. A. Sherif, K. Dreßler, A. N. Ngomo, LIGON – link discovery with noisy oracles, in: P. Shvaiko, J. Euzenat, E. Jiménez-Ruiz, O. Hassanzadeh, C. Trojahn (Eds.), Proceedings of the 15th International Workshop on Ontology Matching (co-located with ISWC), CEUR Workshop Proceedings 2788, 2020, pp. 48–59.

[28] N. Pernelle, F. Saïs, D. Symeonidou, An automatic key discovery approach for data linking, Journal of Web Semantics 23 (2013) 16–30.

[29] D. Symeonidou, V. Armant, N. Pernelle, F. Saïs, SAKey: Scalable Almost Key Discovery in RDF Data, in: Proceedings of 13th International Semantic Web Conference (ISWC), LNCS 8796, Springer, 2014, pp. 33–49.

[30] S. Ferré, Exploring the Application of Graph-FCA to the Problem of Knowledge Graph Alignment, in: P. Cordero, O. Krídlo (Eds.), Proceedings of the Sixteenth International Conference on Concept Lattices and Their Applications (CLA), CEUR Workshop Proceedings 3308, 2022, pp. 79–92.

[31] S. Ferré, P. Cellier, Graph-FCA: An extension of Formal Concept Analysis to knowledge graphs, Discrete applied mathematics 273 (2020) 81–102.

[32] A. Haller, J. D. Fernández, M. R. Kamdar, A. Polleres, What Are Links in Linked Open Data? A Characterization and Evaluation of Links between Knowledge Graphs on the Web, ACM Journal of Data and Information Quality 12 (2) (2020) 9:1–9:34.

[33] A. Haller, A. Polleres, D. Dobriy, N. Ferranti, S. J. R. Méndez, An Analysis of Links in Wikidata, in: P. Groth, M. Vidal, F. M. Suchanek, P. A. Szekely, P. Kapanipathi, C. Pesquita, H. Skaf-Molli, M. Tamper (Eds.), Proceedings of the 19th International Conference on Semantic Web (ESWC), Lecture Notes in Computer Science 13261, Springer, 2022, pp. 21–38.

[34] J. Portisch, N. Heist, H. Paulheim, Knowledge graph embedding for data mining vs. knowledge graph embedding for link prediction – two sides of the same coin?, Semantic Web 13 (3) (2022) 399–422.

[35] E. Yang, F. Hao, Y. Yang, C. D. Maio, A. Nasridinov, G. Min, L. T. Yang, Incremental Entity Summarization With Formal Concept Analysis, IEEE Transactions on Services Computing 15 (6) (2022) 3289–3303.