Uncertainty-sensitive reasoning for inferring sameAs facts in linked data

Mustafa Al-Bakri, Manuel Atencia, Jérôme David, Steffen Lalande, Marie-Christine Rousset

ECAI 2016
G.A. Kaminka et al. (Eds.)

doi:10.3233/978-1-61499-672-9-698

Uncertainty-Sensitive Reasoning for Inferring sameAs Facts in Linked Data

Mustafa Al-Bakri[1,2] and **Manuel Atencia**[1] and **Jérôme David**[1]
Steffen Lalande[2] and **Marie-Christine Rousset**[1,3]

Abstract. Discovering whether or not two URIs described in Linked Data — in the same or different RDF datasets — refer to the same real-world entity is crucial for building applications that exploit the cross-referencing of open data. A major challenge in data interlinking is to design tools that effectively deal with incomplete and noisy data, and exploit uncertain knowledge. In this paper, we model data interlinking as a reasoning problem with uncertainty. We introduce a probabilistic framework for modelling and reasoning over uncertain RDF facts and rules that is based on the semantics of probabilistic Datalog. We have designed an algorithm, ProbFR, based on this framework. Experiments on real-world datasets have shown the usefulness and effectiveness of our approach for data linkage and disambiguation.

1 INTRODUCTION

Linked Data provides access to huge, continuously growing amounts of open data in RDF format that describe properties and links on entities referenced by Uniform Resource Identifiers (URIs). Data interlinking consists in deciding whether two URIs refer to the same real-world entity. This is a crucial task for developing innovative applications on top of Linked Data, that exploit the cross-referencing of data [16, 12]. This task is often referred to as data linkage, but it is also known as record linkage and entity resolution, and it has been widely studied for the case of relational data [9]. As regards Linked Data, data interlinking is especially challenging since (1) tools need to scale well with large amounts of data, (2) data is frequently described using heterogeneous vocabularies (ontologies), and (3) tools need to deal with uncertain data as Linked Data contains data which is inherently incomplete, and very often noisy.

In the context of Linked Data and RDF data, different approaches to data interlinking have been proposed. Most of them are based on numerical methods that use linkage rules to compare property values of resources, using similarity measures to handle noisy data. They conclude weighted sameAs links, from which the links with higher weights are expected (but never guaranteed) to be correct [29, 19]. These approaches suffer from two weaknesses. First, rules cannot be chained, as they are thought to be applied only once; and second, weights are combined in a non-formal manner, since there is no formal semantics that captures the combination of weights. A few other works take a logical approach to data interlinking and use logical rules equipped with full reasoning [24, 2]. They make use of uniqueness constraints (such as inverse functional properties and keys) and other schema constraints, domain knowledge and alignments between different vocabularies which can be modelled as logical rules. They enable rule chaining to infer sameAs links. Logical approaches applying only certain rules over clean and complete data guarantee to provide sound results, i.e., a 100% precision. However, the recall may be low because in Linked Data, data is inherently incomplete and possibly noisy. Input facts may be missing to trigger rules, either because some values for properties involved in rules conditions are absent for some URIs, or because some of these values are noisy with some misspelling that prevents some conditions to be satisfied. In addition, rules may be missing to infer sameAs facts with certainty, although some strong evidence could be obtained from the combination of soft constraints.

This paper introduces a rule-based approach to data interlinking in the context of Linked Data based on uncertain reasoning for inferring sameAs facts. Our contribution is threefold:

- A declarative framework based on probabilistic Datalog [15] in which uncertain facts are modelled as probabilistic facts, and that allows to model in the form of probabilistic rules different kinds of uncertain knowledge useful for inferring sameAs facts.
- An inference algorithm, ProbFR, that takes as input a dataset — possibly including probabilistic facts — and probabilistic rules, and that computes for each of the inferred facts the probability of the fact to be true, as well as the provenance of this computation.
- A series of experiments done with an implementation of ProbFR over three real-world large RDF datasets that show (1) the gain of using uncertain data and knowledge for data interlinking, (2) the gain of using full uncertain reasoning (rule chaining), and (3) the benefits of having probabilities attached to inferred facts for discarding incorrect sameAs links.

There are two main reasons for our choice of probabilistic Datalog as a basis for our approach. First, Datalog rules on top of RDF facts capture in a uniform way most of the OWL and RDFS constraints that are useful for inferring sameAs facts (which includes inverse functional properties and keys), domain knowledge and alignments between different vocabularies. Probabilistic Datalog, in turn, allows to add uncertainty to knowledge simply by attaching probabilistic symbolic events to rules and facts. Uncertain knowledge may be provided by domain experts or may be learnt by specialised automatic tools (as in the case of weighted ontology mappings [11], pseudo keys [6, 28] and complex link specifications [21]).

Second, when compared to other approaches to probabilistic logical reasoning [23, 7], probabilistic Datalog fits better into the setting of Linked Data. These approaches typically make the close-world assumption and perform a supervised learning of probabilistic weights

[1] Univ. Grenoble Alpes, CNRS, Inria, LIG, F-38000 Grenoble, France.
[2] Institut National de l'Audiovisuel, F-94366 Bry-sur-Marne, France.
[3] Institut Universitaire de France, F-75005 Paris, France

that requires full observation of the domain. However, Linked Data makes the open-world assumption and contains very large datasets, which will make these approaches to suffer from scalability issues. In probabilistic Datalog, the uncertain formulas are restricted to Horn rules and ground atoms. Furthermore, probabilities can only be computed for inferred facts. This is a restricted setting compared to Markov Logic or other statistical relational learning. However, the probabilities can be computed more efficiently and more transparently from the provenance expressions that can be obtained for each inferred fact in a forward-chaining manner.

The remainder of the paper is organised as follows. In Section 2 we describe the probabilistic model and the inference algorithm at the core of our approach, and we also make explicit the underlying assumptions for its effectiveness in the setting of Linked Data. In Section 3 we illustrate by example our approach for modelling uncertain data and knowledge useful for data interlinking. Section 4 shows, through experiments conducted on real-world datasets, the feasibility and the added-value of this approach to discover sameAs links. In Section 5, we position our work with respect to existing works, and finally we conclude in Section 6.

2 PROBABILISTIC FRAMEWORK FOR REASONING OVER UNCERTAIN RDF FACTS AND RULES

We have designed a probabilistic framework to model and reason on uncertain RDF facts and rules based on the semantics of probabilistic Datalog [15]. Probabilistic Datalog extends (deterministic) Datalog [1] by associating each ground fact and each instantiated rule with a probabilistic *event* that the corresponding fact or rule is true. Each derived fact is then inferred with its *provenance* in the form of an event expression made of a boolean combination of the events of the ground facts and rules involved in its derivation. It can be written in disjunctive normal form, in which a conjunction of events represents a derivation branch, and disjunctions represent the different derivation branches. Some simplifications can be done before the computation of the resulting probabilities: a conjunction containing disjoint events can be suppressed; events known to be certain can be removed from the conjunctions where they are involved, thus leading to conjunctions with only uncertain events. An extreme case is when a conjunction is made of certain events only, which represents a way to derive a fact with certainty. In this case the whole event expression can be simplified to $\top$ which denotes certain events.

The logical semantics of the (simplified) event expressions is then the basis for computing the probability of the corresponding derived facts in function of the probabilities assigned to the events identifying the input facts and rules taking part in their derivation. In the general case, computing the probability of the disjunction of conjunctions of events requires knowing the probabilities of all the combinations of events in the expression. In practice — and, in particular, in applications dealing with large amounts of data — only the probabilities of single events will be known. We will then make the same default assumptions of independence or disjointness of single events, as it is usually done in most Information Retrieval models [14]. To meet such assumptions, we have to impose some constraints on the rules that will be explained below.

Probabilistic RDF facts extend the standard data model of Linked Data used to state properties on entities referenced by Uniform Resource Identifiers (URIs). Properties are themselves identified by URIs. Data properties relate entities with literals (e.g. numbers, strings or dates), while object properties relate two entities.

A **probabilistic RDF fact** is an RDF triple $t = (s, p, o)$ (in which the subject s is a URI, the predicate p is a URI, and the object o may be either a URI or a literal) associated with an event key e denoting the probabilistic event that t is true.

A **probabilistic RDF rule** is a rule with variables, associated with an event key denoting the probability that any of its instantiations is true. Rules have the form $r : TP_1(v_1) \wedge \ldots \wedge TP_k(v_k) \Rightarrow TP(v)$ where $TP_1(v_1), \ldots, TP_k(v_k)$ and $TP(v)$ are *triple patterns* of the form (s^v, p, o^v) in which the subject s^v or the object o^v may be variables. We consider **safe** rules, i.e. rules such that all the variables in the conclusion are also in the condition part.

Each probabilistic RDF fact and rule are assigned a distinct event key, except the certain facts and rules that are assigned the special event key $\top$ denoting events that are certain. For a probabilistic fact f, we will denote by $e(f)$ the probabilistic event e associated with the fact f. We will write $e(r)$ in the case of a probabilistic rule r.

In rules, we also allow conditions $B(\bar{x}, \bar{a})$ where B is a built-in predicate (i.e. a function call), $\bar{x}$ a vector of variables appearing in the triple conditions of the same rule, and $\bar{a}$ may be a non empty set of values of parameters for calling B. The following rule is an example of a rule with a built-in predicate ($Similar$):

$$r_0 : (?x\ hasName\ ?s_1) \wedge (?y\ hasName\ ?s_2) \wedge Similar(?s_1, ?s_2, levenshtein, 0.2) \Rightarrow (?x\ sameName\ ?y)$$

For each pair of strings (s_1, s_2) for which the two triple conditions are satisfied by the facts $(i_1\ hasName\ s_1)$ and $(i_2\ hasName\ s_2)$, $Similar(s_1, s_2, levenshtein, 0.2)$ applies normalised Levenshstein distance $levenshtein(s_1, s_2)$ on strings s_1 and s_2. If this distance is less than 0.2, it will return the corresponding probabilistic fact $Similar(s_1, s_2, levenshtein, 0.2)$ with $1 - levenshtein(s_1, s_2)$ as probability.

The semantics of a knowledge base $F \cup R$ composed of a finite set of facts F and a finite set of rules R can be given based on the least fixed point of immediate consequence operator T_R defined below.

Definition 1
- $F, R \vdash_1 f$ *iff a rule* $TP_1(v_1) \wedge \ldots \wedge TP_k(v_k) \Rightarrow TP(v)$ *is in* R *and there exists a mapping* θ *from its variables to constants such that* $f = \theta.TP(v)$ *and* $\theta.TP_i(v_i) \in F$ *for every* $i \in [1..k]$.
- $F, R \vdash f$ *iff there exists* i *such that* $f \in T_R(F_i)$ *where* $F_0 = F$ *and for every* $i \geq 0$, $F_{i+1} = T_R(F_i) = F_i \cup \{f | F_i, R \vdash_1 f\}$.

For safe rules, there exists a unique least fixed point F_n, denoted by $SAT(F, R)$, such that for every $k \geq n$, $F_k = T_R(F_n)$, i.e. there exists a step in the iterative application of the immediate consequence operator for which no new fact is inferred. Several forward-chaining algorithms exist to compute $SAT(F, R)$, in particular the semi-naive bottom-up evaluation in Datalog [1], and the RETE algorithm [13] that is implemented in many rule-based reasoners, including in Semantic Web tools such as Jena.[4]

The semantics of inferred probabilistic facts can be obtained based on their *provenance* defined as boolean combinations of all the events associated with the input facts and rules involved in their inference.

Definition 2 *For every fact* f *in* $SAT(F, R)$*, the provenance of* f *(denoted by* $Prov_{R,F}(f)$*) is defined as follows:*

- *if* $f \in F$*, then* $Prov_{R,F}(f) = e(f)$,
- *otherwise, let* $R(f)$ *be the set of instantiated rules* (r, θ) *having* f *as conclusion (i.e. rules* $TP_1(v_1) \wedge \ldots \wedge TP_k(v_k) \Rightarrow TP(v)$ *for which* θ *is a mapping such that* $\theta.TP(v) = f$ *and* $\theta.TP(v_i) \in SAT(F, R)$ *for every* $i \in [1..k]$*). Then:*

[4] https://jena.apache.org/documentation/inference/

$Prov_{R,F}(f) = \bigvee_{(r,\theta)\in R(f)}(e(r) \wedge \bigwedge_{i\in[1..k]} Prov_{R,F}(\theta.TP_i(v_i)))$
For every fact f in $SAT(F,R)$, its probability $P(f)$ is defined as the probability of its provenance: $P(f) = P(Prov_{R,F}(f))$.

Illustrative example. Let us consider the following probabilistic RDF facts and rules (for which we omit to display the event keys) composed of 5 input facts and of 4 rules expressing different ways to infer sameAs facts between individuals (to have the same name, to have the same name and the same birthdate, to be married to the same individual, or by transitivity of the sameAs relation):

$f_1 : (i_1\ sameName\ i_2)$
$f_2 : (i_1\ sameBirthDate\ i_2)$
$f_3 : (i_1\ marriedTo\ i_3)$
$f_4 : (i_2\ marriedTo\ i_3)$
$f_5 : (i_2\ sameName\ i_4)$
$r_1 : (?x\ sameName\ ?y) \Rightarrow (?x\ sameAs\ ?y)$
$r_2 : (?x\ sameName\ ?y), (?x\ sameBirthDate\ ?y) \Rightarrow (?x\ sameAs\ ?y)$
$r_3 : (?x\ marriedTo\ ?z), (?y\ marriedTo\ ?z) \Rightarrow (?x\ sameAs\ ?y)$
$r_4 : (?x\ sameAs\ ?z), (?z\ sameAs\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

Three derived facts are obtained with their provenance:

$Prov_{R,F}((i_1\ sameAs\ i_2)) = (e(r_1) \wedge e(f_1)) \vee (e(r_2) \wedge e(f_1) \wedge e(f_2)) \vee (e(r_3) \wedge e(f_3) \wedge e(f_4))$
$Prov_{R,F}((i_2\ sameAs\ i_4)) = (e(r_1) \wedge e(f_5))$
$Prov_{R,F}((i_1\ sameAs\ i_4)) = e(r_4) \wedge Prov_{R,F}((i_1\ sameAs\ i_2)) \wedge Prov_{R,F}((i_2\ sameAs\ i_4))$

The fact $(i_1\ sameAs\ i_2)$ can be inferred as a result of 3 different derivation branches (one using the rule r_1 and the input fact f_1, one using r_2 and f_1 and f_2, and the third one using r_3 and f_3 and f_4). The second fact $(i_2\ sameAs\ i_4)$ results from a single derivation branch using the rule r_1 and the fact f_5. The last one illustrates how the provenances can be built iteratively during the saturation process: the last derivation step leading to the inference of $(i_1\ sameAs\ i_4)$ involves the rule r_4 and two facts inferred at a previous iteration (namely, $(i_1\ sameAs\ i_2)$ and $(i_2\ sameAs\ i_4)$) for which the provenance must be combined with the event key of r_4.

These provenance expressions can be simplified by exploiting facts and rules that are certain. For instance, if we know that the two facts f_2 and f_3 are certain as well as the rule r_4, we can suppress $e(f_2)$, $e(f_3)$ and $e(r_4)$ in the conjuncts of the above expressions because they are all equal to the event $\top$ always true. We now obtain for $Prov_{R,F}((i_1\ sameAs\ i_2))$:

$(e(r_1) \wedge e(f_1)) \vee (e(r_2) \wedge e(f_1)) \vee (e(r_3) \wedge e(f_4))$

When many facts and several rules are certain, such simplifications lead to a drastic reduction of the size of provenance expressions, which is important for the scalability of the approach in practice.

This example illustrates how the construction and simplification of the provenance can be incorporated into the saturation process and how a given forward-reasoning algorithm can be easily extended to compute the provenance during the inference of corresponding facts.

The ProbFR algorithm. Algorithm 1 describes the ProbFR algorithm that we have implemented and used in our experiments. It starts with the set of initial facts and rules and repeats inference steps until saturation. Each inference step (Line (4) to (15)) triggers all the rules whose conditions can be matched with known facts (i.e. input facts or facts inferred at previous steps). At each iteration, the set Δ contains the facts that have been inferred at the previous iteration. The constraint (expressed in Line (6)) that rules are only triggered if at least one of their conditions can be matched with facts in Δ guarantees that instantiated rules are not triggered twice during the inference process. The algorithm stops as soon as no new fact has been inferred during a given iteration (i.e. Δ_1 remains empty over this iteration). The algorithm returns the set F_{sat} of inferred facts, and computes for each of them an event expression $x(f)$ (Lines (10) and (11)). The function $\mathcal{N}_\vee$ denotes the transformation of a conjunction into its disjunctive normal form. It consists in applying iteratively the distributivity of the conjunction connector ($\wedge$) over the the disjunction connector ($\vee$), and in simplifying when possible the (intermediate) results as follows: (1) remove the duplicate events and the certain events $\top$ from each conjunction of events, (2) if a conjunction within a disjunction becomes empty (i.e. if all its events are certain), replace the whole disjunction by $\top$. Each event expression $x(f)$ is thus $\top$ or of the form $Conj_1 \vee ... \vee Conj_l$ where $Conj_i$ is a conjunction of event keys tracing the uncertain input facts and rules involved into one of the l branches of uncertain derivation of f.

Algorithm 1: The ProbFR algorithm
ProbFR(F, R)
Input: A set F of input (probabilistic) facts and a set R of (probabilistic) rules
Output: The set F_{sat} of inferred (probabilistic) facts with for each inferred fact f its event expression $x(f)$

(1) **for each** $f \in F$: $x(f) \leftarrow e(f)$
(2) $F_{sat} \leftarrow F$
(3) $\Delta \leftarrow F$
(4) **repeat**
(5) $\Delta_1 \leftarrow \emptyset$
(6) **foreach** rule r: $c_1 \wedge \ldots \wedge c_k \Rightarrow c$ for which there exists a substitution θ and facts $f_1, \ldots, f_k \in F_{sat}$ (among which atleast one of them belongs to Δ) such that $f_i = \theta.c_i$ for every $i \in [1..k]$:
(7) let $f = \theta.c$:
(8) **if** $f \notin F_{sat}$
(9) **add** f to Δ_1
(10) $x(f) \leftarrow \mathcal{N}_\vee(e(r) \wedge \bigwedge_{i\in[1..k]} x(f_i))$
(11) **else** $x(f) \leftarrow x(f) \vee$
(12) $\mathcal{N}_\vee(e(r) \wedge \bigwedge_{i\in[1..k]} x(f_i))$
(13) $F_{sat} \leftarrow F_{sat} \cup \Delta_1$
(14) $\Delta \leftarrow \Delta_1$
(15) **until** $\Delta_1 = \emptyset$
(16) **return** F_{sat}

The termination of the ProbFR algorithm is guaranteed because all the rules are safe. The only facts that can be inferred from safe rules and a set F of ground atoms are instantiations of conclusion atoms by constants appearing in F. Their number is finite. More precisely, since the input facts and conclusion atoms are built on binary predicates, the number of constants appearing in the input facts is less than $2 \times |F|$ (at most two distinct constants per input fact), and the number of inferred facts is then less than $4 \times |R| \times |F|^2$ (at most as many predicates in conclusion as rules, and for each of them, at most as many instantiations as pairs of constants).

The following theorem states the soundness and completeness of the algorithm.

Theorem 1 *Let F_{sat} be the result returned by $ProbFR(F,R)$:*
$F_{sat} = SAT(F,R)$.
For each $f \in F_{sat}$, let $x(f)$ be the event expression computed by $ProbFR(F,R)$:
$x(f) \equiv Prov_{F,R}(f)$.

For the first point, we prove by induction on i that each iteration $i \geq 1$ of $ProbFR(F, R)$ computes the set of facts $F_i = T_R(F_{i-1})$ (as defined in Definition 1), and thus $SAT(F, R)$ at the last iteration where the least fixed point reached. For the second point, for a derived fact f, we prove, by induction on the number n of iterations of $ProbFR$ after which no new instantiation of rules can infer f, that $x(f)$ is a disjunctive normal form of $Prov_{F,R}(f)$, and therefore is logically equivalent to it.

As a result of Definition 2 and Theorem 1, it is worth to stress that the probability values of inferred facts are independent from the order in which the rules are triggered to derive them.

As a final remark, like in Datalog, we distinguish the predicates that appear in input facts from the predicates involved in inferred facts. Thus, initial uncertain facts cannot be inferred. They can just take part in the provenance derivation of inferred facts.

Data complexity analysis. We are interested in estimating how the worst-case time complexity of the algorithm depends on the size $|F|$ of the input data, which is the most critical parameter in the setting of Linked Data. The number of iterations of ProbFR is atmost $|F_{sat}|$, which is less than $4 \times |R| \times |F|^2$ as shown just above. At each iteration, in the worst case, the condition part of each rule must be evaluated against the facts, and the event expressions for the provenance of the inferred facts must be computed. Let c the maximum number of conditions per rule. The evaluation of each condition part of each rule can be performed in polynomial time (in fact, in at most $|R| \times |F_{sat}|^c$ elementary steps). So the computation of F_{sat} can be done in polynomial data complexity.

For the computation of the event expressions, the most costly operation is the transformation $\mathcal{N}_\vee$ into disjunctive normal form of conjunctions of the form $e(r) \wedge \bigwedge_{i\in[1..k]} x(f_i)$. The number k of conjunctions is less than the bound c of conditions per rule, and each $x(f_i)$ is a disjunction of at most l conjunctions of event keys, where l is the maximum number of uncertain derivation branches for inferred facts. This parameter l is bounded by b^d where d is the maximal depth of reasoning to infer a fact from F and R, and b is the maximal branching factor of $ground(F, R)$ (which denotes the set of rules triggered during the execution of $ProbFR(F, R)$). Therefore, each call of $\mathcal{N}_\vee$ performs at most $b^{d\times c}$ distributivity operations on conjunctions of at most $|F| + |R|$ event keys. Since the maximal depth of reasoning is the number of iterations of $ProbFR(F, R)$, d can be equal to $|F_{sat}|$. Then, the data complexity of the provenance computation may be exponential in the worst-case. This meets known results on query evaluation in probabilistic databases [27]. Different solutions are possible to circumvent this worst-case complexity, like restricting the form of rules/queries like in [10] or imposing some constraints on the input facts (such as a bounded treewidth in [3]). In practice, in particular if most of the input facts are certain, the size of the event expressions remains small. If all the input facts are certain, the only event keys that can be involved in the event expressions are the ones attached to the uncertain rules. The complexity of the algorithm can be controlled by imposing a practical bound on the number l of conjunctions produced in Line (11). This solution is justified in our setting since the computed probabilities are used to keep only the most probable inferred facts, i.e., the facts that are inferred with a probability greater than a given high threshold. For our experiments, we have limited this number l to be 8.

Effective computation of probabilities of inferred facts from their provenance. For each inferred fact, given its provenance as an event expression in disjunctive normal form, the following formula is the basic theoretical tool to compute its probability:

$$P(A \vee B) = P(A) + P(B) - P(A \wedge B) \quad (1)$$

The recursive application of the above formula for computing the probability of a disjunction of l conjunctions of events $E_1 \vee \ldots \vee E_l$ leads to alternate the subtractions and additions of the probabilities of all the possible conjunctions $E_{j_1} \wedge \ldots \wedge E_{j_i}$. This raises two major issues: first, their number is exponential in l; second, the exact values of all these probabilities is usually not available.

A usual way to circumvent the latter is to make the assumption of independence between events, as it is done in probabilistic databases [27] or in most Information Retrieval models [14]. In our case, however, two rules such that the condition part of one rule is contained in the condition part of the second (like the rules r_1 and r_2 of the example) are obviously not independent. For such rules, we enforce pairwise disjointness by imposing that the more general rule applies only if the more specific rules do not apply. In this way, we are sure that the corresponding dependent events do not appear in any provenance expression computed during the saturation process. To be consistent with the probabilistic setting, we also impose that the probability assigned to the event corresponding to the more specific rule (r_2 in our example) is higher than the one assigned to the event of more general rule (r_1 in our example).

For each pair r, r' with same conclusion (up to variables names), we will write $r \preceq r'$ if condition(r) is contained in condition(r'). Checking $r \preceq r'$ can be done by using any conjunctive query containment algorithm [8] with a complexity independent of the data.

To summarise, we make the assumptions of pairwise *disjointness* between events associated with pairs of rules r, r' such that $r \preceq r'$ and *independence* of the events that are not disjoint. For the effective computation of the probability of an inferred fact f, first, the provenance expressions $x(f) = E_1 \vee \ldots \vee E_l$ computed by ProbFR are simplified by removing each conjunction of events E_i in which an event $e(r)$ appears if there is a conjunction of events E_j ($j \neq i$) such that $e(r')$ appears in E_j and $r \preceq r'$, and, second, the probability of f is computed by iteratively applying the formula (1) on the resulting provenance expression.

In our example, the rules r_1 and r_2 are such that $r_1 \preceq r_2$. We can thus remove the conjuncts containing $e(r_1)$ and we obtain

$x((i_1\ sameAs\ i_2)) = (e(r_2) \wedge e(f_1)) \vee (e(r_3) \wedge e(f_4))$

Now, considering the remaining events as independent, we can compute the effective probability of $P((i_1\ sameAs\ i_2))$ as follows:

$(P(e(r_2)) \times P(e(f_1))) + (P(e(r_3)) \times P(e(f_4))) -$

$(P(e(r_2)) \times P(e(f_1)) \times P(e(r_3)) \times P(e(f_4)))$

Checking, for every two rules r, r', whether r' is more generic than r is done before launching ProbFR, as it is independent from the facts. Then, within ProbFR, at each update (Lines (11), (12)), it is the function $\mathcal{N}_\vee$ that suppresses the conjuncts of $x(f)$ involving more generic rules than r: a simple scan of $x(f)$ makes it possible.

This simplification has an impact on the practical complexity of the effective computation of the probabilities, even if, in theory and in the worst-case, it remains exponential in the number l of remaining conjunctions within provenance expressions. As we have explained it before, this number l can be bounded in practice in the algorithm.

The assumption of disjointness between events associated with rules r, r' such $r \preceq r'$ is important for the feasibility of the approach but it also fits well with the open-world assumption that holds in Linked Data. In fact, it captures a restricted form of negation since, under this disjointness assumption, the event $e(r)$ models worlds where the condition part of r is satisfied and the additional conditions of r' are not satisfied.

Setting up of the input probabilities. The above approach for probabilistic inference is agnostic with respect to the way the input probabilities are obtained, either given by experts, returned by built-in predicates or tools, or learned by supervised methods. This said, it is important to note that training sets (required by supervised machine learning techniques) that would be big enough to scale to the setting of Linked Data do not exist and are almost impossible to build manually. On the other hand, it is quite easy for domain experts to decide whether a given rule is uncertain, but setting up its probability is tricky. The two-steps computation of a provenance-based approach as ours has the big advantage to possibly re-compute the numerical values of probabilities for the inferred facts from the provenance expressions computed once for all. This enables to start with a rough setting of rules probabilities chosen from a small set of values just for distinguishing rules on a simple scale of uncertainty (for instance set at 0.9 the rules a priori considered as almost always certain, 0.8 the rules judged as highly probable but less than the previous ones, and so on), and to adjust these values a posteriori based on a feedback on a sample of results. The provenance of wrong sameAs links inferred with a high probability provides explicitly the rules involved in the different reasoning branches leading to their derivation. It is a useful information for a domain expert to choose the rules to penalize by decreasing their numerical probabilities.

3 MODELING UNCERTAINTY USING PROBABILISTIC RULES AND FACTS

When used for data interlinking, rules typically translate varied knowledge that combines schema constraints, alignments between different ontologies and general properties on OWL relations such as owl:sameAs. This knowledge may be certain, but, very often, it has some degree of uncertainty. It is the case when a correspondence in an ontology alignment is attached a confidence value lower than 1, or when domain experts provide knowledge they are not 100% sure about, or the case of pseudo-keys that are automatically computed by pseudo-key discovery tools [6, 28]. This uncertain knowledge can be translated by means of probabilistic rules.

Tables 1 and 2 show rules translating, respectively, certain and uncertain knowledge for the task of interlinking person entities in DBpedia and MusicBrainz datasets. These rules are actually part of the rules that we used in our experiments (reported in Section 4). Rule musicalArtist in Table 1, for example, is a certain rule that translates the DBpedia knowledge that the class dbo:musicalArtist is subsumed by dbo:Artist. Rule enrich_dboBand1 translates a certain correspondence in an alignment between Schema.org vocabulary and DBpedia ontology stating that the class schema:MusicGroup is subsumed by dbo:Band. The rule sameAsVIAF is a certain rule that translates the assertion that the VIAF id is a key for persons and, therefore, allows to infer sameAs links between person entities from DBpedia and MusicBrainz. Notice that this rule actually involves the two equivalent properties dbp:viaf and mb:ViafID of DBpedia and MusicBrainz vocabularies. This means that the condition ($?x$ dbp:viaf $?id$) in the rule will be instantiated by a DBpedia entity, and ($?y$ mb:ViafID $?id$) by a MusicBrainz entity. This kind of "key across different datasets" is called a link key in the literature [5]. Note also that instead of using owl:sameAs we use our own customised sameAs predicates (:sameAsPerson) which allowed us to easily identify the type of the inferred sameAs links in our experiments. Rule sameAsIsPerson1 is a certain rule that translates transitivity of sameAs.

Rule similarNamesPerson deserves special attention because it contains a built-in predicate (namely MBsolrsimilar) that encapsulates the call to a full-text search tool (namely Solr[5]) to extract strings from MusicBrainz similar to labels of person entities in DBpedia. More precisely, for each string instantiation s of the variable $?l$, obtained by mapping with DBpedia facts the two first conditions ($?x$ rdf:type dbo:Person) and ($?x$ rdfs:label $?l$) of the rule, MBsolrsimilar(s, 0.8, $?z$, 'person_mb') is a procedure call returning as many probabilistic facts MBsolrsimilar(s, 0.8, s', 'person_mb') as labels s' of person entities in MusicBrainz detected by Solr as similar to s with a similarity greater than 0.8. The probability attached to each probabilistic fact MBsolrsimilar(s, 0.8, s', 'person_mb') is the calculated string similarity. Thus similarNamesPerson is a certain rule that will infer uncertain facts of the form ($?x$:solrPSimilarName $?z$) due to condition MBsolrsimilar($?l$,0.8,$?z$,'persons_mb'), which will be instantiated with built-in uncertain facts. Built-in predicates such as MBsolrsimilar enable to embed standard similarity functions into our rule-based approach to overcome the problem of misspelling errors in names of persons, groups and songs that may occur in DBpedia and MusicBrainz datasets.

Table 2 shows three additional rules allowing to infer sameAs links between person entities from DBpedia and MusicBrainz datasets, but, in contrast with the sameAsVIAF rule explained above, they are not 100% certain. Rule sameAsBirthDate, for example, says that if two persons have similar names and the same birthdate then they are $likely$ to be the same person. This rule must be considered uncertain for two reasons. First, it relaxes the strict condition of having exactly the same name by the soft constraint of having similar names as it is specified by ($?x$:solrPSimilarName $?l$). Second, strictly speaking the properties "name" and "birthdate" do not constitute a key, even if it is likely that two named entities representing persons that are well-known enough to be described in datasets like DBpedia and MusicBrainz will refer to the same person if they share the same name and birthdate. In fact, sameAsBirthDate translate a *soft* link key, as it combines the equivalent properties dbo:birthDate and mb:beginDateC that are used in DBpedia and MusicBrainz vocabularies to relate a person with her date of birth. The rules sameAsPersonArtistWr and sameAsMemberOfBand are uncertain too. The first one says that, if two persons have similar names and they are artists of songs with similar names, they are the same person, and the second rule says that if two persons have similar names and are members of musical bands with similar names, they are the same person. Again, this may not be always true, but in most cases. The weights in Table 2 correspond to the probabilistic events associated with each of these uncertain rules.

An important point to emphasise is that the (certain or uncertain) rules allowed in our rule-based modelling express pieces of knowledge that can be assembled and combined through several reasoning steps. For instance, the condition ($?u1$ dbo:artist $?x$) of the sameAsPersonArtistWr rule may be triggered by facts inferred by the musicalArtist rule. The chaining between rules is not known in advance and is determined by the input datasets which they apply to. In addition, due to recursive rules (such as sameAsIsPerson1 rule), even if the termination of the saturation process is guaranteed, the number of reasoning steps cannot be known in advance and also depends on the input datasets. It is worthwhile to note that recursive rules add an expressive power that is required for data linkage in particular to express sameAs transitivity.

The translation into rules can be semi-automatic, for instance for translating into certain rules schema constraints that have been declared in OWL such as the functionality or transitivity of some re-

[5] http://lucene.apache.org/solr/

ID	Conditions	Conclusion
musicalArtist	($?u$ dbo:musicalArtist $?x$)	($?u$ dbo:artist $?x$)
enrich_dboBand1	($?x$ rdf:type schema:MusicGroup)	($?x$ rdf:type dbo:Band)
sameAsVIAF	($?x$ dbp:viaf $?id$), ($?y$ mb:ViafID $?id$)	($?x$:sameAsPerson $?y$)
sameAsIsPerson1	($?x$:sameAsPerson $?y$), ($?z$ mb:is_person $?y$)	($?x$:sameAsPerson $?z$)
similarNamesPerson	($?x$ rdf:type dbo:Person), ($?x$ rdfs:label $?l$), MBsolrsimilar($?l$,0.8,$?z$,'persons_mb')	($?x$:solrPSimilarName $?z$)

Table 1. Certain rules for interlinking person entities in DBpedia and MusicBrainz.

ID	Conditions	Conclusion	Weight
sameAsBirthDate	($?x$:solrPSimilarName $?l$), ($?y$ skos:myLabel $?l$), ($?x$ dbo:birthDate $?date$), ($?y$ mb:beginDateC $?date$)	($?x$:sameAsPerson $?y$)	w_1
sameAsPersonArtistWr	($?u1$ dbo:artist $?x$), ($?u1$:solrWrSimilarName $?lu$), ($?y$ mb:writer $?u2$), ($?u2$ skos:myLabel $?lu$), ($?x$:solrPSimilarName $?lp$), ($?y$ skos:myLabel $?lp$)	($?x$:sameAsPerson $?y$)	w_2
sameAsMemberOfBand	($?x$:solrPSimilarName $?l$), ($?y$ skos:myLabel $?l$), ($?y$ mb:member_of_band $?gr2$), ($?gr2$ skos:myLabel $?lg$), ($?gr1$ dbp:members $?x$), ($?gr1$:solrGrSimilarName $?lg$)	($?x$:sameAsPerson $?y$)	w_3

Table 2. Uncertain rules for interlinking person entities in DBpedia and MusicBrainz.

lations, or for translating into (certain or uncertain) rules alignments discovered by ontology mapping tools [11]. A certain number of uncertain rules useful for data interlinking must however be provided by domain experts to express fine-grained knowledge that may be specific to the datasets concerned by the linkage task. While it is quite easy for domain experts to decide whether a given rule is uncertain, setting up its probability is tricky. The two-steps computation described in Section 2 has the big advantage to allow iterative adjustment of probabilistic weights associated to rules.

In our experiments, such an incremental adjustment for the probabilities of the three uncertain rules of Table 2 resulted into: $w_1 = 0.9$, $w_2 = 0.4$ and $w_3 = 0.6$.

It is worth emphasising that rules with quite low probabilities (such as 0.4 for the sameAsPersonArtistWr rule) can yet significantly contribute to the final probability of a fact inferred by different reasoning branches. For instance, the resulting probability of a fact inferred from (certain facts and) 3 independent rules each with a 0.4 probability is 0.78, and in the case of 4 such rules it raises to 0.87.

4 EVALUATION

We have conducted experiments to evaluate the performance of our method on real datasets. Our main goal was to measure the effectiveness of our method to discover links at large scale, and to assess the expected gain in terms of recall and the loss in precision when using uncertain rules instead of certain rules only. We also wanted to show how the probabilistic weights attached to the links allow to filter out incorrect links. Finally, we aimed at comparing our tool to a state-of-the-art interlinking tool, namely Silk [29].

4.1 Experimental Setting

We used three datasets in our experiments: DBpedia, INA and MusicBrainz. The objective was to find sameAs links between named entities of person, musical band, song and album included in the datasets. Our choice of these datasets was based upon the fact that these are all large datasets (tens of millions of triples), and of a very different nature: DBpedia was built from Wikipedia infoboxes, INA from catalog records mainly containing plain text, and MusicBrainz from more structured data coming from a relational database.

The DBpedia version we used was DBpedia 2015-04,[6] the latest version at the time the experiments were conducted. From all available (sub) datasets, we only used the ones including RDF triples with properties appearing in the rules that we used in the experiments (below we give more details about the rules), which make together one single dataset of around 73 million RDF triples. The INA dataset contains around 33 million RDF triples, while the MusicBrainz dataset around 112 million RDF triples. The INA dataset was built from all the records (plain text) in a catalog of French TV musical programs using an specialised RDF extractor. Some RDF facts in the INA dataset have numerical weights between 0 and 1 since their accuracy could not be 100% assessed during the extraction process. The MusicBrainz dataset[7] was built from the original postgreSQL table dumps available at the MusicBrainz web site using an RDF converter. This version is richer than the one of the LinkedBrainz project.[8]

Table 3 shows the number of person, musical band, song and album entities in each of the considered datasets, where Person, e.g. symbolises the class union of all the classes that represent persons in each dataset. No bands or albums are declared in INA, written NA (not applicable) in Table 3.

Class	DBpedia	MusicBrainz	INA
Person	1,445,773	385,662	186,704
Band	75,661	197,744	NA
Song	52,565	448,835	67,943
Album	123,374	1,230,731	NA

Table 3. Number of person, musical band, song and album entities in DBpedia, MusicBrainz and INA.

We have designed two sets of rules that we used as inputs for our algorithm to interlink DBpedia and MusicBrainz first and then MusicBrainz and INA. We came up with 86 rules for interlinking DBpedia and MusicBrainz, from which 50 of them are certain and 36 are

[6] http://wiki.dbpedia.org/Downloads2015-04
[7] Available at http://exmo-web.inrialpes.fr/MusicBrainz
[8] http://linkedbrainz.org/

uncertain, and 147 rules for interlinking MusicBrainz and INA, 97 of them certain and 50 uncertain.[9] By a way of example, Table 1 and Table 2 of Section 3 include some of the certain and uncertain rules that we used for interlinking DBpedia and MusicBrainz.

ProbFR has been implemented on top of Jena RETE and uses SWI-Prolog v6 to compute the disjunctive normal forms for the event expressions during RETE inference. Prolog is also used to implement the second step of ProbFR, i.e. to compute effective probabilities given event expressions. In order to avoid potential combinatorial explosion, the current parameter of ProbFR is tuned to a maximum of 8 derivation branches for each event expression. All ProbFR experiments were run on a Bi-processor intel Xeon 32 x 2.1GHz, 256 GB of RAM, with Linux CentOS 6 as operating system.

4.2 Experimental Results

We ran our algorithm to interlink DBpedia and MusicBrainz first, and then MusicBrainz and INA, using in each case the corresponding rules. Our algorithm discovered 144,467 sameAs links between entities of DBpedia and MusicBrainz and 28,910 sameAs links between entities of MusicBrainz and INA. Additionally, our algorithm found 132,166 sameAs links internal to the INA dataset.

In order to evaluate the quality of the found links, and since no gold standard was available, we estimated precision, recall and F-measure by sampling and manual checking. In order to compute precision, for each of the classes considered we took a sample of 50 links from the links found by our algorithm (i.e. 200 links in total for DBpedia and MusicBrainz, and 100 links for MusicBrainz and INA), and we manually checked whether these links were correct. For computing recall, we randomly selected 50 instances of each of the classes, and we found links manually. Then, we calculated recall based on this make-do gold standard. F-measure was based on the estimations of precision and recall.

In order to assess the gain of using uncertain rules, we also ran our algorithm only with certain rules, and then we compared the results obtained using only certain rules with the ones obtained using all rules (both certain and uncertain rules). This concerned the experiments between DBpedia and MusicBrainz only, as no other certain rule than sameAs transitivity was used for MusicBrainz and INA.

Table 4 shows all the results. Let us focus on the results concerning DBpedia and MusicBrainz. As expected, when certain rules were used only, precision was 100%. This only concerns Person and Band classes because the initial set of rules did not include any certain rule concluding links for Song and Album (written NA in Table 4). However, recall was very low: 0.08 for Person and 0.12 for Band. When both certain and uncertain rules were used, a 100% precision was achieved for Person and Album classes only, since for Band and Song, precision was 0.94 and 0.96, respectively. However, recall increased significantly for Person and Band: 0.80 and 0.84. This shows the gain of using uncertain rules for data linkage. Now, when looking at the samples of Band and Song classes, we realised that all wrong links had a probability value lower than 0.9 and 0.6, respectively. This means that, when limited to those links having a probability value higher or equal to 0.9 and 0.6, the estimated precision for the classes Band and Song was 100% (Table 5). The estimated recall was 0.80 and 0.54. This shows the gain of using weights for interlinking.

Table 6 shows the number of links that are discovered when n sameAs rules[10] are implied in the derivation. For instance, 28,614 links are discovered using two sameAs rules, and among these links 27,692 are new links, i.e. they were not discovered using only one rule. With tools like Silk and LIMES, using the same set of rules, we can expect to find around 115,609 links only.

4.3 Comparison with Silk

Since Silk cannot handle rule chaining, we divided the rules used by ProbFR into sameAs rules (i.e. rules with sameAs in the conclusion), and intermediate rules that are used to trigger antecedents of other rules (including the sameAs rules). We manually translated these intermediate rules into SPARQL Update queries and these updates were performed before the Silk execution. Some sameAs rules could not be translated into Silk because they are recursive (sameAs appears in their antecedent and conclusion). To be able to compare methods on the same basis, we employed the levenshtein normalised distance with a threshold of 0.2, which corresponds to the similarity parameter set up to 0.8 in Solr. The aggregation of different comparisons within a rule was performed using maximum distance to be compliant with the conjunction used in rules. We executed Silk for interlinking DBpedia and MusicBrainz. Silk found 101,778 sameAs links, from which 100,544 were common to the ones found by ProbFR. ProbFR found 43,923 links that were not discovered by Silk and Silk found 1,234 links not discovered by ProbFR. In theory all the links discovered by Silk should have been discovered by ProbFR and Silk should have found up to 115,609 links. These differences can be explained by the way levensthein distance are implemented in each tools and by a normalisation of URL that is performed by ProbFR and not available in Silk. As a conclusion, ProbFR outperformed Silk because of rule chaining (more links are discovered). Dealing with uncertainty allows to enhance precision without losing much recall.

In terms of time performance, Silk took more than 53 hours (with 16 threads, blocking activated, on a Bi-processor Intel Xeon, 24 x 1.9GHz) while ProbFR achieved the task in 18 hours (on a Bi-processor Intel Xeon, 32 x 2.1GHz). Even if the difference could be partially explained by the difference in hardware, the main reason comes from implementation design. Silk mainly relies on disk indexing and uses few RAM (around 1-2 GB) while ProbFR runs into main memory and uses around 250 GB of RAM for this experiment.

5 RELATED WORK

There exists a considerable number of systems that (semi) automatically perform data linkage [12]. Most of these approaches consists in applying a set of linkage rules that produce links. These linkage rules specify which properties and how their values are compared. The comparison can be strict or based on some similarity measure between property values of two entities. Linkage rules can be defined manually by a domain expert or learned from data. There are numerous works on learning linkage rules. Some methods are supervised like [18, 20], others are unsupervised like [22, 21]. Our work focuses on generating links given a set of rules and how to infer rules from data is out of the scope of this paper.

Tools like Silk [29] and LIMES [19] are designed to efficiently compare similarities between values of all or some of entities properties and aggregate them. They did not consider reasoning with rules. Silk specifications can be translated into logical rules with built-in functions for computing and aggregating similarity degrees between

[9] All the rules can be found at http://exmo-web.inrialpes.fr/probfr/rules_INA_MB.txt and http://exmo-web.inrialpes.fr/probfr/rules_DBpedia_MB.txt

[10] We only consider rules that conclude to sameAs statements because other rules can be handled with preprocessing by tools like Silk or LIMES.

	DBpedia and MusicBrainz						MusicBrainz and INA					
	Only certain rules			All rules			Only certain rules			All rules		
	P	R	F	P	R	F	P	R	F	P	R	F
Person	1.00	0.08	0.15	1.00	0.80	0.89	NA	NA	NA	1.00	0.34	0.51
Band	1.00	0.12	0.21	0.94	0.84	0.89	NA	NA	NA	NA	NA	NA
Song	NA	NA	NA	0.96	0.74	0.84	NA	NA	NA	1.00	0.40	0.57
Album	NA	NA	NA	1.00	0.53	0.69	NA	NA	NA	NA	NA	NA

Table 4. Precision (P), recall (R) and F-measure (F) for the task of interlinking DBpedia and MusicBrainz datasets, and MusicBrainz and INA datasets, using certain rules only, and certain and uncertain rules together.

	P	R	F
$\text{Band}_{\geqslant 0.90}$	1.00	0.80	0.89
$\text{Song}_{\geqslant 0.60}$	1.00	0.54	0.72

Table 5. Gain of using weights for interlinking DBpedia and MusicBrainz.

# rules	# links	# new links
1	115,609	115,609
2	28,614	27,692
3	1,790	1,152
4	59	14

Table 6. Number of links discovered when n rules are implied in the derivation. Results given for interlinking DBpedia and MusicBrainz.

property values. However, these rules are restricted to linkage rules that are applied independently to each other. The possible chaining between rules is not handled by Silk, which makes it incomplete for the task of discovering all the sameAs links that can be logically inferred. Thus, as it has been pointed out in our experiments, neither Silk nor LIMES [19] (similar to Silk in its principles) would discover sameAs links obtained by transitivity.

L2R [24], Hogan et al. [17] and Al-Bakri et al. [2] handle logical rules and provide full reasoning algorithms that guarantee to infer all the links that can be logically entailed from the rules and facts given as input, either based on forward reasoning for L2R [24], Hogan et al. [17] or on backward reasoning like in Al-Bakri et al. [2]. However, the rules considered in these works are considered to be certain. In LN2R [25], the logical rule-based method of L2R is completed by a similarity-based method (called N2R) applied to pairs of entities for which L2R failed to infer links with certainty.

Dedupalog [4] is a Datalog-like language that has been specially designed for handling constraints useful for record linkage. It handles both hard and soft rules that define respectively valid clusterings and their costs. The associated algorithm computes a valid clustering with a minimal cost. Whereas the general problem is NP-complete, they provide a practical algorithm that scales to the ACM database that contains 436,000 records. Even if the algorithmic techniques are very different from ours, the scalability is obtained by similar restrictions on the rule language. However, the goal is to compute a valid clustering and not to compute probabilities of inferred facts.

Probabilistic logical frameworks such as Markov logic [26] and Probabilistic Soft Logic (PSL) [7] have been used for entity resolution. Markov Logic allows for full probabilistic reasoning. The weights attached to formulas are learned either from data or from probabilities arbitrarily given. This learning phase is made under closed-world assumption. Once a Markov Logic Network is learned, the weighted satisfiability of any candidate link has to be computed. This is not scalable in practice. Then, candidate pairs are filtered using a cheap similarity such as TF.IDF: non matching pairs are added as false atoms. Experiments have been conducted on Cora dataset (1295 instances) and a sample of Bibserv ($10,000$ instances). PSL allows probabilistic inference based on similarities functions. As Markov Logic, formulas' weights are learned making closed world assumption. Furthermore, it allows to assign weights to facts using the similarity of sets of property values (which assumes that sets are fully known). Like Datalog, it is restricted to conjunctive rules. Experiments have been performed on the task of Wikipedia article classification and ontology matching.

Contrary to aforementioned approaches, in ProbFR, probability computation and inference are separated. All rules are iteratively applied to compute the saturation and the provenances of every deduced facts. Probabilities are then computed from the provenances. This allows to change the probabilities assigned to rules and reevaluated quickly the probabilities of inferred facts without recomputing the saturation. Another difference is that probabilities attached to formulas can be given or learned from data. No further learning is required.

6 CONCLUSIONS

In this paper, we have shown that it is possible to capture and exploit in a uniform rule-based framework knowledge that may be uncertain but very useful for data linkage. For this, we have adapted the formal setting of Probabilistic Datalog [15] in a way well-suited to Linked Data in which data is inherently incomplete and possibly noisy. Our experiments have shown that this approach is feasible in practice and brings important gain in terms of recall compared to purely logical approaches based on rules and facts that are supposed to be 100% true, while keeping good precision.

Decoupling the symbolic computation of provenances from the numerical computation of probabilities makes probabilistic reasoning more modular and more transparent for users. This provides explanations on probabilistic inference for end-users, and useful traces for experts to set up the input probabilistic weights.

Currently, the threshold for filtering the probabilistic sameAs facts that will be retained as being true must be set up and adjusted manually. As future work, we plan to design a method to set up this threshold automatically by, besides inferring sameAs facts, inferring differentFrom facts too, and then exploiting the sameAs and differentFrom facts (and their probabilities) that are inferred for the same pairs of entities. We also plan to design a backward-reasoning algorithm able to deal with probabilistic rules, that could be combined with the ProbFR probabilistic forward-reasoner for importing on demand useful data from external sources.

ACKNOWLEDGEMENTS

This work has been partially supported by the ANR projects Pagoda (12-JS02-007-01) and Qualinca (12-CORD-012), the joint NSFC-ANR Lindicle project (12-IS01-0002), and LabEx PERSYVAL-Lab (11-LABX-0025-01).

REFERENCES

[1] Serge Abiteboul, Richard Hull, and Victor Vianu, *Foundations of Databases*, Addison-Wesley, 1995.

[2] Mustafa Al-Bakri, Manuel Atencia, Steffen Lalande, and Marie-Christine Rousset, 'Inferring same-as facts from linked data: an iterative import-by-query approach', in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pp. 9–15. AAAI Press, (2015).

[3] Antoine Amarilli, Pierre Bourhis, and Pierre Senellart, 'Provenance circuits for trees and treelike instances', in *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part II*, volume 9135 of *Lecture Notes in Computer Science*, pp. 56–68. Springer, (2015).

[4] Arvind Arasu, Christopher Ré, and Dan Suciu, 'Large-scale deduplication with constraints using dedupalog', in *Proceedings of the 25th International Conference on Data Engineering, ICDE 2009, March 29 2009 - April 2 2009, Shanghai, China*, pp. 952–963. IEEE Computer Society, (2009).

[5] Manuel Atencia, Jérôme David, and Jérôme Euzenat, 'Data interlinking through robust linkkey extraction', in *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, pp. 15–20. IOS Press, (2014).

[6] Manuel Atencia, Jérôme David, and François Scharffe, 'Keys and pseudo-keys detection for web datasets cleansing and interlinking', in *Knowledge Engineering and Knowledge Management - 18th International Conference, EKAW 2012, Galway City, Ireland, October 8-12, 2012. Proceedings*, volume 7603 of *LNCS*, pp. 144–153. Springer, (2012).

[7] Matthias Bröcheler, Lilyana Mihalkova, and Lise Getoor, 'Probabilistic similarity logic', in *UAI 2010, Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence, Catalina Island, CA, USA, July 8-11, 2010*, pp. 73–82. AUAI Press, (2010).

[8] A.K. Chandra and P.M. Merlin, 'Optimal implementation of conjunctive queries in relational databases', in *Proceedings of the 9th ACM Symposium on Theory of Computing*, pp. 77–90, (1975).

[9] Peter Christen, *Data Matching - Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*, Data-Centric Systems and Applications, Springer, 2012.

[10] Nilesh Dalvi and Dan Suciu, 'The dichotomy of probabilistic inference for unions of conjunctive queries.', *Journal of the ACM*, **59**(6), 17–37, (2012).

[11] Jérôme Euzenat and Pavel Shvaiko, *Ontology Matching, Second Edition*, Springer, 2013.

[12] Alfio Ferrara, Andriy Nikolov, and François Scharffe, 'Data linking for the semantic web', *International Journal on Semantic Web and Information Systems*, **7**(3), 46–76, (2011).

[13] Charles Forgy, 'Rete: A fast algorithm for the many patterns/many objects match problem', *Artificial Intelligence*, **19**(1), 17–37, (1982).

[14] Norbert Fuhr, 'Probabilistic models in information retrieval', *The Computer Journal*, **3**(35), 243–255, (1992).

[15] Norbert Fuhr, 'Probabilistic datalog: implementing logical information retrieval for advanced applications', *Journal of the American Society for Information Science*, **51**(2), 95–110, (2000).

[16] Tom Heath and Christian Bizer, *Linked Data : Evolving the Web into a Global Data Space*, Morgan and Claypool, 2011.

[17] Aidan Hogan, Antoine Zimmermann, Jürgen Umbrich, Axel Polleres, and Stefan Decker, 'Scalable and distributed methods for entity matching, consolidation and disambiguation over linked data corpora', *Journal of Web Semantics*, **10**, 76–110, (2012).

[18] Robert Isele and Christian Bizer, 'Active learning of expressive linkage rules using genetic programming', *Web Semantics: Science, Services and Agents on the World Wide Web*, **23**(0), (2013).

[19] Axel-Cyrille Ngonga Ngomo and Sören Auer, 'LIMES - A time-efficient approach for large-scale link discovery on the web of data', in *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pp. 2312–2317. IJCAI/AAAI, (2011).

[20] Axel-Cyrille Ngonga Ngomo and Klaus Lyko, 'EAGLE: efficient active learning of link specifications using genetic programming', in *The Semantic Web: Research and Applications - 9th Extended Semantic Web Conference, ESWC 2012, Heraklion, Crete, Greece, May 27-31, 2012. Proceedings*, volume 7295 of *Lecture Notes in Computer Science*, pp. 149–163. Springer, (2012).

[21] Axel-Cyrille Ngonga Ngomo and Klaus Lyko, 'Unsupervised learning of link specifications: deterministic vs. non-deterministic', in *Proceedings of the 8th International Workshop on Ontology Matching co-located with the 12th International Semantic Web Conference (ISWC 2013), Sydney, Australia, October 21, 2013.*, volume 1111 of *CEUR Workshop Proceedings*, pp. 25–36. CEUR-WS.org, (2013).

[22] Andriy Nikolov, Mathieu d'Aquin, and Enrico Motta, 'Unsupervised learning of link discovery configuration', in *The Semantic Web: Research and Applications - 9th Extended Semantic Web Conference, ESWC 2012, Heraklion, Crete, Greece, May 27-31, 2012. Proceedings*, Lecture Notes in Computer Science, pp. 119–133. Springer, (2012).

[23] Matthew Richardson and Pedro M. Domingos, 'Markov logic networks', *Machine Learning*, **62**(1-2), 107–136, (2006).

[24] Fatiha Saïs, Nathalie Pernelle, and Marie-Christine Rousset, 'L2R: A logical method for reference reconciliation', in *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*, pp. 329–334. AAAI Press, (2007).

[25] Fatiha Saïs, Nathalie Pernelle, and Marie-Christine Rousset, 'Combining a logical and a numerical method for data reconciliation', *Journal on Data Semantics*, **12**, 66–94, (2009).

[26] Parag Singla and Pedro M. Domingos, 'Entity resolution with markov logic', in *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), 18-22 December 2006, Hong Kong, China*, pp. 572–582. IEEE Computer Society, (2006).

[27] Dan Suciu, Dan Olteanu, Christopher Ré, and Chistoph Koch, *Probabilistic Databases*, Morgan & Claypool, 1995.

[28] Danai Symeonidou, Vincent Armant, Nathalie Pernelle, and Fatiha Saïs, 'Sakey: Scalable almost key discovery in RDF data', in *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*, volume 8796 of *LNCS*, pp. 33–49. Springer, (2014).

[29] Julius Volz, Christian Bizer, Martin Gaedke, and Georgi Kobilarov, 'Silk - A link discovery framework for the web of data', in *Proceedings of the WWW2009 Workshop on Linked Data on the Web, LDOW 2009, Madrid, Spain, April 20, 2009.*, volume 538 of *CEUR Workshop Proceedings*. CEUR-WS.org, (2009).