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

doi:10.3233/978-1-61499-672-9-1017

Markov Logic Networks with Numerical Constraints

Melisachew Wudage Chekol and **Jakob Huber** and **Christian Meilicke** and **Heiner Stuckenschmidt**[1]

Abstract. Markov logic networks (MLNs) have proven to be useful tools for reasoning about uncertainty in complex knowledge bases. In this paper, we extend MLNs with numerical constraints and present an efficient implementation in terms of a cutting plane method. This extension is useful for reasoning over uncertain temporal data. To show the applicability of this extension, we enrich log-linear description logics (DLs) with concrete domains (datatypes). Thereby, allowing to reason over weighted DLs with datatypes. Moreover, we use the resulting formalism to reason about temporal assertions in DBpedia, thus illustrating its practical use.

1 Motivation

Recent advances in data mining and information extraction have paved the way for the automatic construction of knowledge bases (KBs) from different sources, for instance, the NELL KB [26]. Often, the extraction tools used to construct such KBs produce weighted (or probabilistic) facts, due to the awareness that the implemented techniques cannot guarantee the completeness, correctness or consistency of the generated facts. Moreover, a large fraction of these facts can be temporal and may contain concrete data values, for example dates, times, latitudes/longitudes, numerical values measured in different units, and so on. Besides, the set of rules used to consolidate the KB will also be a set of weighted and unweighted rules with datatypes (numerical constraints). In this work, we propose an extension of Markov logic networks (MLNs) [32] that supports reasoning on numerical constraints.

We extend an MLNs inference approach (cutting plane [33]) to handle numerical constraints. This is difficult because numerical constraints may be infinite or continuous and reasoning about comparisons of two constraints is difficult to handle natively in MLNs. As an example, consider the following hard constraints which express: (1) a person whose age is between 13 and 19 is a teenager, and (2) a valid life span of a person is between 0 and 150 years:

$$1\ \forall x, y{:}Person(x) \wedge age(x, y) \wedge y \geq 13 \wedge y \leq 19 \Rightarrow Teen(x)$$
$$2\ \forall p, bd, dd{:}bdate(p, bd) \wedge ddate(p, dd) \Rightarrow lifeSpan(bd, dd),$$
$$lifeSpan(bd, dd) = ((dd - bd) > 0) \wedge ((dd - bd) \leq 150)$$

To the best of our knowledge, MLNs do not support numerical constraints (like the rules in the above example). Therefore, we propose an extended cutting plane approach to MLNs inference to introduce just the necessary constraint violations for the current MAP hypothesis taking also violated numerical constraints into account.

The proposed extension is useful for reasoning over uncertain temporal KBs as well as weighted Horn DLs with concrete domains. In this paper, we show that this extension is powerful enough to reason over log-linear description logics (DLs) with concrete domains. Log-linear DLs [27] have been proposed as a way to combine tractable DLs with uncertainty reasoning while preserving the semantics of the DL part. They have also proven to be useful in situations where conflicting information from different sources has to be integrated into a coherent model. Successful applications include ontology matching [31] and information fusion [27]. So far the applicability of the combined model has been limited by the fact that it only supported reasoning on the schema level. In our work, we extend log-linear DLs to overcome its restriction to schema-level reasoning by introducing instances and numerical concrete domains (datatypes).

In DL, a concrete domain (CD) is a construct that can be used to define new classes by specifying constraints on attributes that have literal values (as opposed to relationships to other abstract entities). For instance, the axiom $Teenager \equiv Person \sqcap \geq_{13}(age) \sqcap \leq_{19}(age)$ defines a teenager as a person whose age is at least 13 and at most 19. While CDs, also referred to as *datatypes*, are a well-studied construct in classical DL (see for instance [22, 24]), this is barely the case in the probabilistic or log-linear extensions of DLs that have been proposed more recently (e.g. [7, 16, 11, 27, 21, 6, 35]. The contribution of this paper is the following:

- an extension of MLN with numerical constraints (MLN_{NC}),
- an extension of log-linear $\mathcal{EL}$ with nominals and CDs by making use of MLN_{NC},
- an application of MLN_{NC} for reasoning about a specific subdomain in the DBpedia knowledge base that contains numerical datatypes.
- an application of MLN_{NC} for debugging weighted temporal KBs.

Outline: In the next section we present the preliminaries: log-linear models, the lightweight ontology language $\mathcal{EL}^{++}$, and log-linear $\mathcal{EL}$ obtained by combining log-linear models with $\mathcal{EL}^{++}$. In order to reason over uncertain temporal KBs, in Section 3, we extend MLNs with numerical constraints. In order to demonstrate additional benefits of this framework, we (1) extend log-linear $\mathcal{EL}$ with CDs (in Section 4 and Section 5), and (2) present experimental results on debugging temporal KBs and computing the MAP state of an uncertain KB which contains datatypes (Section 6). Finally, we summarize the related work in Section 6, before making concluding remarks in Section 7.

2 Preliminaries

In this section, we present a brief summary of: Log-linear models, $\mathcal{EL}^{++}$, and log-linear $\mathcal{EL}$. For a detailed discussion on these subjects, we refer the reader to [2, 32, 34, 27] and the references therein.

[1] Data and Web Science Group, University of Mannheim, 68161 Mannheim, Germany, email: mel|jakob|christian|heiner@informatik.uni-mannheim.de

2.1 Log-Linear Models

A *log-linear model* is a compact representation of a probability distribution over assignments to a set of discrete-valued random variables $\mathcal{X} = \{X_1, \ldots, X_n\}$ [17, 9]. The log-linear model is defined in terms of a set of feature functions $F = \{f_1(\mathbf{X}_1), \ldots, f_k(\mathbf{X}_k)\}$, each of which is a function that defines a numerical value for each assignment $\mathbf{x}_k$ to some subset $\mathbf{X}_k \subseteq \mathcal{X}$. Given a set of feature functions F, the parameters of the log-linear model are weights $W = \{w_k : f_k \in F\}$. The overall distribution is then defined as: $P(\mathbf{x}) = Z^{-1}\exp\big(\sum_{f_k \in F} w_k f_k(\mathbf{x}_k)\big)$, where $\mathbf{x}_k$ is the assignment to $\mathbf{X}_k$ within $\mathbf{x}$, and Z is the normalization constant. A log-linear model induces a *Markov network* over $\mathcal{X}$, where there is an edge between every pair of variables X_i, $X_j \in \mathbf{X}_k$ that appear together in some feature $f_k(\mathbf{X}_k)$ and $\mathbf{X}_k$ is a clique. Markov networks can also be encoded as a log-linear models by defining a feature function for every assignment of variables $\mathbf{x}_c$ to a clique $\mathbf{X}_c$.

Markov Logic Networks (MLNs) can be seen as a first-order template language for log-linear models with binary variables. MLNs combine Markov networks and first-order logic (FOL) by attaching weights to first-order formulas and viewing these as templates for features of Markov networks [32]. An MNL L is a set of pairs (F_i, w_i) where F_i is a formula in FOL and w_i is a real number representing a weight. Together with a finite set of constants C, it defines a Markov Network $M_{L,C}$, where $M_{L,C}$ contains one node for each possible grounding of each predicate appearing in L. The value of the node is 1 if the ground predicate is true, and 0 otherwise. The probability distribution over possible worlds x specified by the ground Markov network $M_{L,C}$ is given by:

$$P(X = x) = \frac{1}{Z}\exp\Big(\sum_{i=1}^{F} w_i n_i(x)\Big)$$

where F is the number of formulas in the MLN and $n_i(x)$ is the number of true groundings of F_i in x. The groundings of a formula are formed simply by replacing its variables with constants in all possible ways. The *Herbrand base* HB is the set of all ground predicates (atoms) that can be constructed using the predicates in L and the constants in C. Each subset of the Herbrand base is a *Herbrand interpretation* specifying which ground atoms are true. A Herbrand interpretation H is a Herbrand model of L, written $\models_H L$, iff it satisfies all groundings of formulas in L.

There are two principal reasoning tasks in MLN, namely, MAP inference and Marginal inference. MAP inference is the task of finding the most probable world given some observations also referred to as evidence. Given the observed variables $E = e$, the MAP problem aims to find an assignment of all non-evidence (hidden) variables $X = x$ such that $\mathbf{I} = \text{argmax}_x P(X = x \mid E = e)$. We denote by $\mathbf{I}$, the assignment x which leads P to be maximal, i.e., a MAP state. In order to compute a MAP state of an MLN, the problem can be formulated as an integer linear program (ILP) using the cutting plane inference algorithm.

2.2 $\mathcal{EL}^{++}$

$\mathcal{EL}^{++}$ is the DL underlying the OWL 2 profile OWL-EL[2].

Syntax: Given a set of concept names $\mathsf{N_C}$, role names $\mathsf{N_R}$, individuals $\mathsf{N_I}$, and feature names $\mathsf{N_F}$, $\mathcal{EL}^{++}$ concepts and roles are formed according to the syntax given in Table 1. A concept in $\mathcal{EL}^{++}$ is either a top, bottom concept, an atomic concept, a concrete concept or a complex concept (formed by conjunction and existential restriction). A *concrete domain* D is a pair $(\Delta^{\mathsf{D}}, \mathsf{P}^{\mathsf{D}})$ with Δ^{D} a set and P^{D} a set of predicate names. Each $p \in \mathsf{P}$ is associated with an arity $n > 0$ and an extension $p^{\mathsf{D}} \subseteq (\Delta^{\mathsf{D}})^n$. The abstract and concrete domains are linked via a set of *feature names* $\mathsf{N_F}$. In Table 1, p denotes a predicate of some concrete domain D_j and $f_1, \ldots, f_n$ are feature names. In this work, we consider only numerical CDs. However, our approach can easily be extended to handle other CDs. An $\mathcal{EL}^{++}$ TBox contains a set of GCI (General Concept Inclusion) axioms, i.e., $C \sqsubseteq D$, as well as (RI) role inclusion axioms, i.e., $r_1 \circ r_2 \sqsubseteq r$.

Semantics: The semantics of $\mathcal{EL}^{++}$ concepts and roles, shown in Table 1, is given by an interpretation function $\mathcal{I} = (\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ which consists of a non-empty (abstract) domain $\Delta^{\mathcal{I}}$ and a mapping $.^{\mathcal{I}}$ that assigns to each atomic concept $A \in \mathsf{N_C}$ a subset of $\Delta^{\mathcal{I}}$, to each abstract role $R \in \mathsf{N_R}$ a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, to each concrete relation $f \in \mathsf{N_F}$ a subset of $\Delta^{\mathcal{I}} \times \mathsf{D}$, and to each individual $a \in \mathsf{N_I}$ an element of $\Delta^{\mathcal{I}}$. The mapping $.^{\mathcal{I}}$ is extended to all concepts and roles as shown in Table 1.

Table 1. The $\mathcal{EL}^{++}$ with concrete domains.

Name	Syntax	Semantics
top	$\top$	$\Delta^{\mathcal{I}}$
bottom	$\bot$	$\emptyset$
nominal	$\{a\}$	$\{a^{\mathcal{I}}\}$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
existential restriction	$\exists r.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y \in \Delta^{\mathcal{I}} : (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
concrete domain	$p(f_1, \ldots, f_n)$ for $p \in \mathcal{P}^{\mathsf{D}_j}$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y_1, \ldots, y_k \in \Delta^{\mathsf{D}_j} : f_i^{\mathcal{I}}(x) = y_i \text{ for } 1 \leq i \leq k \wedge (y_1, \ldots, y_k) \in p^{\mathsf{D}_j}\}$
GCI	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
RI	$r_1 \circ r_2 \sqsubseteq r$	$r_1^{\mathcal{I}} \circ r_2^{\mathcal{I}} \subseteq r^{\mathcal{I}}$
concept assertion	$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
role assertion	$r(a,b)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$

Knowledge about specific objects can be expressed using concept and role assertions of the form $C(a)$ and $R(a,b)$. The axioms and assertions are contained in the TBox and ABox, respectively, which together form a knowledge base (KB). An $\mathcal{EL}^{++}$ knowledge base $KB = (\mathcal{T}, \mathcal{A})$ consists of a set $\mathcal{T}$ of general concept inclusion axioms (TBox) and role inclusion axioms, and possibly a set $\mathcal{A}$ of assertional axioms (ABox). Despoina et al [8] have extended the inference/completion rules of $\mathcal{EL}^{++}$ for the concrete domains by exploiting the notion of safety which keeps tractability of reasoning while enhancing expressivity. We will make use of these rules to provide datatype reasoning in log-linear $\mathcal{EL}$.

To simplify the translation of weighted $\mathcal{EL}^{++}$ KBs into FOL, we first obtain the *normal* form of the KB in such a way that satisfiability is preserved [2, 18]. An $\mathcal{EL}^{++}$ KB is in *normal* form $\tau(\mathsf{KB})$ if its axioms are in the following form:

$$C(a) \qquad r(a,b) \qquad A \sqsubseteq C \quad A \sqcap B \sqsubseteq C$$

[2] http://www.w3.org/TR/owl2-profiles/

$\exists r.A \sqsubseteq C \qquad A \sqsubseteq \exists r.B \qquad r_1 \sqsubseteq r_2 \quad r_1 \circ r_2 \sqsubseteq r$
$A \sqsubseteq p(f_1, \ldots, f_n) \quad p(f_1, \ldots, f_n) \sqsubseteq B$

where $A, B, C \in \mathsf{N_C} \cup \{\top\}$ and $C \in \{\bot\}$; $r, r_1, r_2 \in \mathsf{N_R}$; $f_1, \ldots, f_n \in \mathsf{N_F}$; and $a, b, c \in \mathsf{N_I}$. For a finite set $\mathsf{N} \subseteq \mathsf{N_C} \cup \mathsf{N_R}$ of concept and role names the *set of all normalized axioms constructible from* N is the union of (a) all normalized GCIs constructible from concept and role names in N and the top and bottom concepts; and (b) all normalized RIs constructible from role names in N.

Axioms of the form $C \sqsubseteq \{o\}$ are shown to be problematic for practical reasoning in the $\mathcal{EL}$ family with nominals. However, experiments have shown that such axioms are rarely found in real world ontologies [14, 15]. In addition, by restricting their usage efficient practical reasoning can be achieved. Furthermore, for safe $\mathcal{EL}^{++}$, the inference rules given in [2] and proved in [15] are sound and complete. Therefore, for this study, we consider safe nominals as defined below.

Definition 1 (Kazakov et al [15]) *An $\mathcal{EL}^{++}$ concept C is* safe *if C has only occurrences of nominals in subconcepts of the form $\exists R.\{o\}$; C is* negatively safe *if C is either safe or a nominal. A GCI $C \sqsubseteq D$ is* safe *if C is negatively safe and D is safe. An $\mathcal{EL}^{++}$ ontology is* safe *if all its concept inclusions are safe.*

For instance, the axioms $\{a\} \sqsubseteq C$ and $\{a\} \sqsubseteq \exists r.\{b\}$ are safe. In this study, we consider safe $\mathcal{EL}^{++}$. It is possible to provide a probabilistic extension of safe $\mathcal{EL}^{++}$ using MLNs. In which, a safe $\mathcal{EL}^{++}$ KB can be seen as a set of hard constraints on the set of possible interpretations: if an interpretation violates even one axiom or assertion, it has zero probability. The basic idea in MLNs is to soften these constraints, i.e., when an interpretation violates one axiom or assertion in the KB it is less probable, but not impossible. The fewer axioms an interpretation violates, the more probable it becomes. Each axiom and assertion has an associated weight that reflects how strong a constraint is: the higher the weight, the greater the difference in log probability between an interpretation that satisfies the axiom and one that does not, other things being equal [32].

2.3 Log-Linear $\mathcal{EL}$

Log-linear $\mathcal{EL}$ (LogEL) is an extension of $\mathcal{EL}^{++}$ with log-linear models that enables reasoning over uncertain $\mathcal{EL}^{++}$ TBoxes [27].

Syntax: A LogEL knowledge base $\mathsf{KB} = (\mathsf{KB}^{\mathsf{D}}, \mathsf{KB}^{\mathsf{U}})$ consists of a deterministic knowledge base KB^{D} and an uncertain knowledge base KB^{U} with $\mathsf{KB}^{\mathsf{D}} \not\models_{\mathcal{L}} \bot_{\mathcal{L}}$ and $\mathsf{KB}^{\mathsf{D}} \cap \mathsf{KB}^{\mathsf{U}} = \emptyset$ for some logic $\mathcal{L}$ that supports a valid entailment relation $\models_{\mathcal{L}}$ and notion of contradiction $\bot_{\mathcal{L}}$. The uncertain KB is defined as $\mathsf{KB}^{\mathsf{U}} = \{\langle c_i, w_{c_i}\rangle\}$ where c_i is an axiom or assertion in $\mathcal{L}$ and w_{c_i} is a real-valued weight assigned to c_i. The syntax of an axiom (resp. assertion) is similar to the underlying logic where an uncertain axiom (resp. assertion) has an associated weight as $\{\langle c_i, w_{c_i}\rangle\}$.

Semantics: The semantics of a LogEL knowledge base is based on joint probability distributions over the uncertain KB. Formally, for a given log-linear $\mathsf{KB} = (\mathsf{KB}^{\mathsf{D}}, \mathsf{KB}^{\mathsf{U}})$ and some KB' over the same signature, the probability of KB' is defined as:

$$P(\mathsf{KB}') = \begin{cases} \frac{1}{Z} \exp\left(\sum_{\{(c_i, w_{c_i}) \in \mathsf{KB}^{\mathsf{U}} : \mathsf{KB}' \models c_i\}} w_{c_i}\right) \\ \qquad \text{if } \mathsf{KB}' \not\models_{\mathcal{L}} \bot_{\mathcal{L}} \wedge \mathsf{KB}' \models_{\mathcal{L}} \mathsf{KB}^{\mathsf{D}}, \\ 0 \qquad \text{otherwise} \end{cases}$$

where Z is the normalization constant of the log-linear probability distribution P. Note that in MAP inference (i.e., obtaining the most probable KB) Z is not computed.

Example 1 *Consider the following uncertain LogEL axioms: (1) a researcher is (probably) someone who published something, and (2) a famous researcher influenced (probably) someone who in turn has influenced someone else.*

(1) $\langle Researcher \sqsubseteq \exists published.\top, 0.8\rangle$
(2) $\langle FamousResearcher \sqsubseteq \exists influenced.(\exists influenced.\top) \sqcap Researcher, 0.6\rangle$

A LogEL KB can be normalized into an equivalent KB, thus, it can be mapped into first order predicates using a function φ as follows:

$$\begin{array}{cc} A \sqsubseteq C \mapsto \mathrm{sub}(A, C) & \exists r.A \sqsubseteq C \mapsto \mathrm{rsub}(A, r, C) \\ A \sqcap B \sqsubseteq C \mapsto \mathrm{int}(A, B, C) & r_1 \sqsubseteq r_2 \mapsto \mathrm{psub}(r_1, r_2) \\ A \sqsubseteq \exists r.B \mapsto \mathrm{rsup}(A, r, B) & r_1 \circ r_2 \sqsubseteq r \mapsto \mathrm{pcom}(r_1, r_2, r) \end{array}$$

The predicates in this listing are *typed*, r, r_1, r_2 are role names, A and B are concept names, and C is a concept name or the bottom concept ($\bot$). Note that φ is incomplete given an LogEL as defined above. In this work we will extend φ in order to deal with both ABox assertions and CDs (i.e., $p(f_1, \ldots, f_n)$).

The translation of KB^{D} and KB^{U} results in unweighted (hard) and weighted first order formulas respectively. The hard formulas are used, together with the TBox completion rules (F_1) – (F_9) to enforce $\mathsf{KB}' \not\models_{\mathrm{EL}} \bot_{\mathrm{EL}}$ and $\mathsf{KB}' \models_{\mathrm{EL}} \mathsf{KB}^{\mathsf{D}}$ for any possible TBox KB'. Computing the MAP state of a LogEL knowledge base will always result in the most probable non contradictory subset of KB^{U} that entails the previously known axioms KB^{D}.

(F_1) $\mathrm{sub}(c, c)$
(F_2) $\mathrm{sub}(c, \top)$
(F_3) $\mathrm{sub}(c, c') \wedge \mathrm{sub}(c', d) \Rightarrow \mathrm{sub}(c, d)$
(F_4) $\mathrm{sub}(c, c_1) \wedge \mathrm{sub}(c, c_2) \wedge \mathrm{int}(c_1, c_2, d) \Rightarrow \mathrm{sub}(c, d)$
(F_5) $\mathrm{sub}(c, c') \wedge \mathrm{rsup}(c', r, d) \Rightarrow \mathrm{rsup}(c, r, d)$
(F_6) $\mathrm{rsup}(c, r, d) \wedge \mathrm{sub}(d, d') \wedge \mathrm{rsub}(d', r, e) \Rightarrow \mathrm{sub}(c, e)$
(F_7) $\mathrm{rsup}(c, r, d) \wedge \mathrm{psub}(r, s) \Rightarrow \mathrm{rsup}(c, s, d)$
(F_8) $\mathrm{rsup}(c, r_1, d) \wedge \mathrm{rsup}(d, r_2, e) \wedge \mathrm{pcom}(r_1, r_2, r_3) \Rightarrow \mathrm{rsup}(c, r_3, e)$
(F_9) $\neg\mathrm{sub}(c, \bot)$

The above formulas (collectively denoted by $\mathbf{F}$) are universally quantified over all variables. They are partially derived from the completion rules of $\mathcal{EL}^{++}$ [2]. $\top$ and $\bot$ are constant symbols representing the top and bottom concepts. Note that rule F_9 does not belong to the completion rules for $\mathcal{EL}^{++}$. This rule takes the notion of incoherence into account. An incoherent ontology is an ontology that contains an unsatisfiable concept, i.e., a concept that is subsumed by $\bot$. Usually, an unsatisfiable concept indicates that the ontology contains a contradictory set of axioms. An incoherent ontology is not necessarily inconsistent, Thus, we added rule F_9 which allows us to extend the notion of contradiction $\bot_{EL}$ from inconsistency to incoherence. For more technical details on applying the principle of log-linear logic to OWL-EL, we refer the reader to [27]. We will extend these completion rules in order to deal with ABox assertions and CDs.

3 MLN with Numerical Constraints

We extend MLN with numerical constraints resulting in a formalism denoted $\mathsf{MLN_{NC}}$. The constraints are predicates of the form $\theta \bowtie \psi$, where θ and ψ denote variables, numerical constants or algebraic expressions (that might contain elementary operators), and $\bowtie$ is a binary operator which returns a truth value under some grounding.

Definition 2 (MLN with Numerical Constraints ($\mathsf{MLN_{NC}}$)) *A numerical constraint* NC *is composed of numerical constants (such as elements of* $\mathbb{N}$, $\mathbb{I}$, *and so on), variables, elementary operators or functions (such as,* $+$, $*$, $-$, $\div$, $\%$, $\sqrt{\ }$*), standard relations (*$>$, $<$, $=$, $\neq$, $\geq$, $\leq$*), and boolean operators (*$\wedge$, $\vee$, $\neg$*). An* $\mathsf{MLN_{NC}}$ *is a set of pairs* (FC_i, w_i) *where* FC_i *is a formula in FOL that may contain a* NC *and* w_i *is a real number representing the weight of formula* FC_i.

Example 2 *Using* $\mathsf{MLN_{NC}}$ *it is possible to represent the hard constraint: people born before 1850 are probably not alive:* $\{\forall a, y : person(a) \wedge born(a, y) \wedge \mathsf{NC}(y) \Rightarrow dead(y), \mathsf{NC}(y) = y < 1850\}$.

MAP Inference in $\mathsf{MLN_{NC}}$. A common inference task over MLNs is finding the most probable state of the world, i.e., finding a complete assignment to all ground atoms which maximizes the probability. A MAP query corresponds to an optimization problem with linear constraints and a linear objective function. Hence, it can be formulated and solved as an instance of an integer linear program (ILP) using the cutting plane approach, proposed in [34] and extended in [29]. Consequently, we extend the approach in [30] to compute a MAP state for a $\mathsf{MLN_{NC}}$ knowledge base. Cutting plane inference (CPI) operates between the grounding algorithm and the ILP solver. Instead of immediately adding one linear constraint for each ground clause to the ILP formulation, the ILP is initially formulated so as to enforce the evidence to hold in any solution. Based on the solution of this more compact ILP, one determines the violated constraints, adds these to the ILP, and solves the ILP again. This process is repeated until no constraints are violated by an intermediate solution. To elaborate, at the beginning of each CPI iteration it is necessary to determine the violated ground clauses G that are specified by the MLN and are in conflict with the intermediate solution. A binary ILP variable $x_\ell \in \{0, 1\}$ gets assigned to each grounded predicate occurring in a violated clause $g \in \mathsf{G}$. The value of the variable x_ℓ is 1 if the respective literal ℓ is true and 0 if it is false. These variables are used to generate ILP constraints that are added to the ILP for each violated ground clause. For each clause $g \in \mathsf{G}$, we define $L^+(g)$ as the set of ground atoms that occur unnegated in g and $L^-(g)$ as the set of ground atoms that occur negated in g. The transformation scheme depends on the weight $w_g \in \mathbb{R}$ of the violated clause g. It is also necessary to create a binary variable z_g for every g with $w_g \neq \infty$ that is used in the objective of the ILP. For every ground clause g with $w_g > 0$, the following constraint has to be added to the ILP.

$$\sum_{\ell \in L^+(g)} x_\ell + \sum_{\ell \in L^-(g)} (1 - x_\ell) \geq z_g$$

A ground atom ℓ that is set to false (or true if it appears negated) by evidence will not be included in the ILP as it cannot fulfill the respective constraint. For every g with weight $w_g < 0$, we add the following constraint to the ILP:

$$\sum_{\ell \in L^+(g)} x_\ell + \sum_{\ell \in L^-(g)} (1 - x_\ell) \leq (|L^+(g)| + |L^-(g)|) z_g$$

Algorithm 1 Extended cutting planes algorithm

Input: G: ground clauses; E: evidence clauses
Input: $\mathsf{G_{NC}}$: ground clauses with NC
Input: $L^+_{\mathrm{NC}} \cup L^-_{\mathrm{NC}}$: ground (positive or negated) numerical literals
Output: $H^{(t)}$: MAP state

```
1:  procedure ComputeMAP(G, E)
2:      G_ILP ← E ∪ {literal(g) ∈ G and w_g > 0}
3:      Initial solution H^(0) ← all atoms in G_ILP
4:      ILP ← intoILP(∀g ∈ G_ILP)
5:      t ← 0
6:      repeat
7:          G_new ← ∅
8:          for g ∈ G \ G_ILP do
9:              if ((w_g > 0 or w_g = ∞) and H^(t) ⊭ g)
10:                 or (w_g < 0 and H^(t) ⊨ g) then
11:                 if g ∈ G_NC and g contains g_p and
12:                     ((comp(g_p) = 1 and g_p ∈ L^-_NC) or
13:                     (comp(g_p) = 0 and g_p ∈ L^+_NC)) then
14:                     add g \ g_p to G_new
15:                 else   add g to G_new
16:                 end if
17:             end if
18:         end for
19:         if G_new ≠ ∅ then
20:             t ← t + 1
21:             ILP ← intoILP(∀g ∈ G_new)
22:             add g ∈ G_new to G_ILP
23:             H^(t) ← solution of ILP
24:         end if
25:     until G_new = ∅ return H^(t)
26: end procedure
```

The variable z_g expresses if a ground formula g is true considering the optimal solution of the ILP. However, for every g with weight $w_g = \infty$ this variable can be replaced with 1 as the respective formula cannot be violated in any solution:

$$\sum_{\ell \in L^+(g)} x_\ell + \sum_{\ell \in L^-(g)} (1 - x_\ell) \geq 1$$

Finally, the objective of the ILP sums up the weights of the (satisfied) ground formulas:

$$\max \sum_{g \in \mathcal{G}} w_g z_g.$$

The MAP problem can be solved as an ILP problem. Thus, the MAP state corresponds to the solution of the ILP in the last CPI iteration. It can be directly obtained from the solution as the assignment of the variables x_ℓ can be directly mapped to the optimal truth values for the ground predicates, i.e., $x_i = \texttt{true}$ if the corresponding ILP variable is 1 and $x_i = \texttt{false}$ otherwise.

We have extended the cutting planes algorithm proposed by Riedel [33] and optimized in [34] to the truth value of numerical predicates on-demand during each CPI iteration. This extension is sketched in Algorithm 1. The first initial MAP hypothesis is the set $H^{(0)}$. It contains all evidence clauses and ground clauses (containing only a single literal) with positive weights. These clauses have been added to the ILP and are denoted by $\mathsf{G_{ILP}}$. At each CPI iteration, the violated constraints are identified and added to $\mathsf{G_{new}}$. Thus, depending on the numerical predicates and the weight of the ground clauses, the violated constraints are determined (Lines 9–13). Consequently, for

every ground clause $g \in \mathsf{G} \setminus \mathsf{G}_{\mathsf{ILP}}$, if g is not satisfied by the current hypothesis $H^{(t)}$ and $w_g > 0$ or $w_g = \infty$, or g is satisfied and $w_g < 0$, we test if g contains a NC. If g contains a NC (i.e., $g \in \mathsf{G}_{\mathsf{NC}}$) and the numerical predicate g_p is in g, then we compute the numerical constraints corresponding to the predicate g_p with the current grounding, i.e., $comp(g_p)$. If $comp(g_p)$ is *true* and g_p is a negated literal $g_p \in L^-_{\mathrm{NC}}$ or if $comp(g_p)$ is *false* and g_p is a positive literal $g_p \in L^+_{\mathrm{NC}}$, we remove the numerical predicate from g and add g to $\mathsf{G}_{\mathsf{new}}$. Note that we do not introduce ILP variables for numerical predicates as they will not be added to the $\mathsf{G}_{\mathsf{new}}$, hence, to the ILP. Note also that our approach computes the truth value of numerical predicates on-demand, i.e., only when g is not satisfied by the current hypothesis. If g does not contain a numerical predicate, we add g to $\mathsf{G}_{\mathsf{new}}$. If $\mathsf{G}_{\mathsf{new}}$ is empty, then the current hypothesis is optimal and we return it as MAP state. Otherwise, it iterates until an optimal solution is found.

Theorem 1 *Given a ground* $\mathsf{MLN}_{\mathsf{NC}}$ *network, a set of observed variables E and a set of hidden variables X, an interpretation* $\mathbf{I}$ *is a MAP state iff*

$$\mathbf{I} = \operatorname*{argmax}_X P(X \mid E) = \operatorname*{argmax}_X \sum_i w_i n_i(E, X).$$

Proof 1 *(sketch) It is proved that the MAP problem in MLN can be reduced to ILP using the CPI approach [34]. Likewise, we reduce* $\mathsf{MLN}_{\mathsf{NC}}$*'s MAP inference into a maximization problem in ILP using cutting planes. In Algorithm 1, if* G *contains no ground numerical predicates, then the MAP inference in* $\mathsf{MLN}_{\mathsf{NC}}$ *coincides with MAP inference in MLN. Otherwise, it is possible to compute a MAP state by transforming* G *into ILP and integrating the truth value of the numerical predicates to the ILP by computing it outside the CPI setting as shown in Algorithm 1.*

4 Extending LogEL with Concrete Domains

We extend LogEL with uncertain knowledge expressed through numerical CDs.

Definition 3 *A numerical concrete domain* D *is a pair* $(\Delta^{\mathsf{D}}, \mathsf{P}^{\mathsf{D}})$ *with* $\Delta^{\mathsf{D}} \in \{\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}\}$ *and* P^{D} *a set of predicate names. Each* $p \in \mathsf{P}$ *is associated with an arity n and an extension* $p^n \subseteq (\Delta^{\mathsf{D}})^n$.

4.1 Unary concrete domains

We call a CD D *unary* if all predicates $p \in \mathsf{P}^{\mathsf{D}}$ are unary, i.e., $n = 1$. In this section, we consider the unary predicates $\{<_v, \leq_v, >_v, \geq_v, =_v\}$ where $v \in \mathbb{R}$. The DL underlying LogEL admits safeness of CDs to retain tractability of reasoning. We present a unary CD R inspired by safe numerical CDs introduced in [8]. *Safeness* is defined in terms of the position in which a predicate appears with respect to the subsumption ($\sqsubseteq$) relation in an axiom. If a predicate p appears on the right side of $\sqsubseteq$, then we call it a *positive predicate* ($p \in \mathsf{P}^{\mathsf{D}}_+$) and if on the left side, then it is a *negative predicate* ($p \in \mathsf{P}^{\mathsf{D}}_-$).

Definition 4 *A safe unary numerical CD* R *is a triple* $(\Delta^{\mathsf{R}}, \mathsf{P}^{\mathsf{R}}_+, \mathsf{P}^{\mathsf{R}}_-)$ *with* $\Delta^{\mathsf{R}} \subseteq \mathbb{R}$ *a set and* $\mathsf{P}^{\mathsf{R}}_+, \mathsf{P}^{\mathsf{R}}_- \subseteq \{<_v, \leq_v, >_v, \geq_v, =_v\}$ *are sets of* positive *and* negative *predicate names respectively.*

In the following, we consider an extension of LogEL with a safe unary numerical CD R and refer to it as LogEL(R).

Example 3 *We extend Example 1 with axioms that use safe numerical CDs where* $<_{1850} \in \mathsf{P}^{\mathsf{D}}_-$ *and* $<_{1800} \in \mathsf{P}^{\mathsf{D}}_+$.

$$\langle University \sqcap <_{1850}(foundedIn) \sqsubseteq DistinguishedUniversity, 0.4\rangle$$
$$\langle OldUniversity \sqsubseteq <_{1800}(foundedIn) \sqcap University, 0.6\rangle$$

Inference involving axioms that contain the CD R can be done according to the following deduction/completion rules. In Definition 6, we will transform them into FOL formulas for reasoning over LogEL(R) KB.

$$\frac{A \sqsubseteq B \quad B \sqsubseteq p_v(f)}{A \sqsubseteq p_v(f)}$$
$$\frac{A \sqsubseteq p_v(f) \quad p'_{v'}(f) \sqsubseteq B \quad eval(p, v, p', v')}{A \sqsubseteq B}$$

where $p_v \in \mathsf{P}^{\mathsf{D}}_+, p'_{v'} \in \mathsf{P}^{\mathsf{D}}_-$ and $v, v' \in \mathbb{R}$. In addition, the $eval$ function checks if all possible values of the first *operator-value* pair (p, v) are covered by the possible values of the second *operator-value* pair (p', v'), when so, it evaluates to true otherwise false. The function $eval$ is defined based on the domain $\mathbb{R}$ and algebraic operators in R. *Some* of the inequalities that are computed using the $eval$ function are listed below:

$eval(=, v, =, v') := v = v'$
$eval(=, v, >, v') := v > v'$

Example 4 *From the axioms of Example 3, it is possible to infer that* $OldUniversity \sqsubseteq DistinguishedUniversity$ *because* $eval(<, 1800, <, 1850) = 1800 \leq 1850 = true$.

In Section 5, we will see that the computation of the $eval$ function integrates well with the proposed extension of the cutting planes algorithm.

4.2 Binary concrete domains

Unary CDs are not expressive enough to enable to describe the *relationship* between concrete values [22]. For instance, to describe a person that passed away at birth, we can use a binary CD as: $DeadAtBirth \sqsubseteq =(birthDate, deathDate)$. Next, we introduce such expressive binary CDs. The main advantage of binary CDs over unary ones is that the former provides for a richer set of predicates [23]. Consequently, we introduce a CD T which is based on real numbers and a set of unary and binary predicates. T is defined as follows: $\mathsf{T} = (\Delta^{\mathsf{T}}, \mathsf{P}^{\mathsf{T}})$ where $\Delta^{\mathsf{T}} = \mathbb{R}$ and P^{T} is defined as the smallest set containing the following predicates:

- unary predicates $\top_{\mathsf{T}}$ with $(\top_{\mathsf{T}})^{\mathsf{T}} = \mathbb{R}$ and $\bot_{\mathsf{T}}$ with $(\bot_{\mathsf{T}})^{\mathsf{T}} = \emptyset$,
- a unary predicate p_r for each $p \in \{<, \leq, =, \neq, \geq, >\}$ and each $r \in \mathbb{R}$ with $(p_r)^{\mathsf{T}} = \{r' \in \mathbb{R} \mid r'\ p\ r\}$, and
- binary predicates $p \in \{<, \leq, =, \neq, \geq, >\}$ with $(p)^{\mathsf{T}} = \{(r, r') \in \mathbb{R}^2 \mid r\ p\ r'\}$.

We will use the c T for temporal reasoning over weighted LogEL(T) KBs as shown in Section 6.2. In order to reason over such KBs, we design a set of domain specific rules which are used either for inferring a new knowledge or checking conflicts (aka. debugging). This is partly due to most of the current KBs do not contain TBox axioms that involve binary CDs. Note that it is possible to define a temporal CD based solely on Allen's interval relations which are often used in temporal reaoning [1]. Allen's relations describe the relationship between any two intervals over some temporal structure (for instance, a set of time points under a strict total ordering relation $\prec$) and can

be defined in terms of the interval endpoints. And thus, they can be expressed via the binary predicates of the CD T. Hence, the CD T is more expressive than a CD obtained by using Allen relations.

For the binary CDs we do not provide TBox completion rules, instead for the purpose of experimentation, we design a set of KB specific constraints. Examples of such rules are the following:

$$(1)\ \forall x,t1,t2 : Person(x) \wedge birthDate(x,t1) \wedge graduationDate(x,t2) \Rightarrow <(t1,t2) \quad 0.9$$

$$(2)\ \forall x,y,t1 : YoungAuthor(x) \wedge age(x,t1) \Rightarrow <(t1,20) \quad 0.8$$

These rules are used in the experiments as discussed in Section 6.2.

5 Translating LogEL(R) and LogEL(T) into $\mathsf{MLN_{NC}}$

As shown in previous work, the MAP state for a LogEL KB can be computed using a MLN that represents the first order transformation of the KB and entailment rules.

We have already presented the function φ for the translation of LogEL TBox axioms into FOL predicates. We now extend this function to map the CDs and ABox assertions of a LogEL KB into ground FOL predicates. Since the DL, $\mathcal{EL}^{++}$, underlying LogEL is equipped with nominals, ABox assertions can be converted into TBox axioms. Thus, with nominals, the ABox becomes syntactic sugar as shown below:

$$C(a) \Leftrightarrow \{a\} \sqsubseteq C,\ \ r(a,b) \Leftrightarrow \{a\} \sqsubseteq \exists r.\{b\}$$

Instance checking in turn is directly reducible to subsumption checking in the presence of nominals. Thereby reducing ABox reasoning into TBox reasoning. Thus, in LogEL(R) and LogEL(T), we transform the ABox assertions into TBox axioms using nominals. We also extend the definition of normal forms for LogEL KBs with CDs as follows: for a LogEL $\mathsf{KB} = (\mathsf{KB^D}, \mathsf{KB^U})$, its normal form is given by: $\tau'(\mathsf{KB}) = \tau(\mathsf{KB^D}) \cup \bigcup_{(c_i,w_{c_i})\in \mathsf{KB^U}} \tau(c_i)$.

Definition 5 *φ translates a normalized LogEL(R) and LogEL(T) KB's ABox and CDs into FOL formulas in $\mathsf{MLN_{NC}}$ as follows:*

$$\begin{array}{lcl} C(a) & \mapsto & \mathrm{sub}(\{a\}, C) \\ r(a,b) & \mapsto & \mathrm{rsup}(\{a\}, r, \{b\}) \\ A \sqsubseteq p_v(f) & \mapsto & \mathrm{sub}_+(A, p, v, f) \\ p_v(f) \sqsubseteq B & \mapsto & \mathrm{sub}_-(p, v, f, C) \end{array}$$

Definition 6 *The translation of LogEL(R) TBox completion rules into FOL formulas $\mathbf{F}$ in $\mathsf{MLN_{NC}}$ is given below:*

$$(F_{10})\quad \mathrm{sub}(a,b) \wedge \mathrm{sub}_+(b,p,v,f) \Rightarrow \mathrm{sub}_+(a,p,v,f)$$

$$(F_{11})\quad \mathrm{sub}_+(a,p,v,f) \wedge \mathrm{sub}_-(p',v',f,b) \wedge \mathrm{eval}(p,v,p',v') \Rightarrow \mathrm{sub}(a,b)$$

Note that the FOL formulas $\mathbf{F}$ corresponding to the translation of the completion rules are hard constraints. As the translation function φ is bijective, a possible world of $\mathbf{F}$ and a given KB can be turned into a classified and consistent $\mathcal{EL}^{++}(\mathsf{R})$ KB and vice versa as shown in the following lemma.

Lemma 1 *Let KB be a normalized LogEL(R) (resp. LogEL(T)) knowledge base over a finite set N of concept, role, and feature names and let HB be the Herbrand base of $\mathbf{F}$ with respect to N. If $\mathsf{KB}' \subseteq \mathsf{KB}$ is a classified and coherent knowledge base, then $\varphi(\mathsf{KB}')$ is a Herbrand model of $\mathbf{F}$. If $\mathrm{HB}' \subseteq \mathrm{HB}$ is a Herbrand model of $\mathbf{F}$ then $\varphi^{-1}(\mathrm{HB}')$ is a classified and coherent knowledge base.*

For a finite set of concept, role, individual and feature names F, every normalized KB over the signature N which is classified and coherent, corresponds to exactly one Herbrand model of $\mathbf{F}$.

MAP Inference Computing the MAP state of a given LogEL(R) (resp. or LogEL(T)) KB requires to translate the KB with the function φ (see Definition 5) to the equivalent Markov logic formalization. Then the completion rules of Definition 6 are added to this translation. The MAP state is computed with the help of Algorithm 1 applied to this input data. To do so, the evidence clauses $\varphi(\mathsf{KB})$ and the grounding of $\mathbf{F}$ with respect to $\varphi(\mathsf{KB})$ are given as input to the algorithm. Besides, Algorithm 1 uses the external *comp* function whenever one of the grounding of the completion rules (F_{10}) and (F_{11}) requires to check a concrete grounding of the *eval* predicate. Note that the *comp* function can be easily implemented for the specified set of unary and binary numerical predicates (corresponding to a particular CD). Moreover, further extension of the proposed approach will also benefit from the fact that the externally defined *comp* function can be easily extended to deal with other predicates, for example, to support string comparisons. Applying the inverse translation function φ^{-1} to the MAP state, yields the most probable, classified, and consistent $\mathcal{EL}^{++}(\mathsf{R})$ (resp. $\mathcal{EL}^{++}(\mathsf{T})$) KB.

Theorem 2 *Given the following:*

- *a LogEL(R) (resp. LogEL(T)) normalized $\mathsf{KB} = (\mathsf{KB^D}, \mathsf{KB^U})$ over a finite set N of concept, role, individual and feature names,*
- *the Herbrand base HB of the formulas $\mathbf{F}$ with respect to N,*
- *the set of ground formulas G_1 constructed from $\mathsf{KB^D}$, and*
- *the set of ground formulas G_2 constructed from $\mathsf{KB^U}$.*

The most probable coherent and classified ontology is obtained with:

$$\varphi^{-1}(I) = \underset{\mathrm{HB}\supseteq I \models G_1 \cup F}{\arg\max} \left(\sum_{(g_j,w_j)\in G_2 : I \models_{\#} g_j} w_j \right)$$

From Theorem 2 and the results in [27], the problem of computing the most probable, classified and coherent LogEL(R) (resp. LogEL(T)) KB is NP-hard.

6 Experiments

In the following we report about two types of experiments: temporal reasoning in LogEL(R) and LogEL(T) KBs.

6.1 Reasoning in LogEL(R)

In this experiment, we first illustrate in how far our formalism can be applied to a scenario where we reason with a knowledge base $\mathsf{KB} = (\mathsf{KB^D}, \mathsf{KB^U})$ with $\mathsf{KB^U} = \emptyset$. In the second set of experiments we add weights to all assertions in $\mathsf{KB^D}$, while adding at the same time a set of weighted erroneous assertions. We show that we achieve similar reasoning results in this noisy setting, while automatically detecting and removing erroneous numerical assertions.

6.1.1 Deterministic Temporal Reasoning

We apply our method to a specific subdomain of DBpedia that deals with researchers, their alma mater, their birth and death date, their publications together with the publishing date, and the influence relationships between researchers. For our experiments we worked with

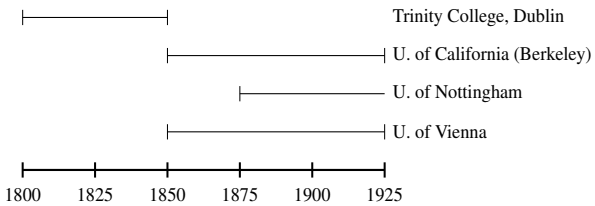

Figure 1. A part of the elite universities from 1800 to 1925.

a slightly simplified signature. For example, we used the object property *influenced* for all assertions using one of the DBpedia properties influenced, academic advisor, and notable student. We extracted a dataset from DBpedia containing >11K persons, >1K universities, >10K books, and ≈10K birth and death dates, >19K influenced assertions, and >700 publishing dates.
Then we added the axiom $FamousResearcher \equiv \exists influenced.(\exists influenced.\top)$ as well as a set of axioms for which the following axiom is an example.

$$\exists hasStudent.(FamousResearcher \sqcap >_{1830}(born) \sqcap \exists authorOf.<_{1910}(published)) \sqsubseteq EliteUniversity_{1850-1900}$$

The first axiom defines a famous researcher as someone has influenced someone who in turn has influenced someone else. The second axiom states that an elite or distinguished university is one which had a student who was a famous researcher born after 1830 and published some work before 1910. With the help of these two axioms we can entail which academic institutions had a significant impact on the research community during the period 1850–1900. We created the same type of axioms for all 50 year periods starting in 1700 using a step size of 25 years. Then we computed the MAP state and retrieved the instances for all $EliteUniversity$ concepts. A subset of the results of our query, with respect to the periods in the range from 1800 to 1925, is shown in Figure 1. Note that the number of $EliteUniversity$ increases over time, which might also be caused by the fact that DBpedia is more complete with respect to more recent publications. Nevertheless, all of the universities from the results obtained are well known and distinguished universities.

6.1.2 Uncertain Temporal Reasoning

In this set of experiments we generated erroneous assertions specifying the publishing date of a book and the birth and death dates of persons. We injected a fraction of 20%, 40%, 60%, 80%, and finally 100% incorrect statements to the dataset. For instance, injecting 20% erroneous facts means that we added 20% additional wrong assertions for each of the three datatype properties. We randomly assigned weights in the range $[0.5, 1.0]$ to the injected assertions and $[0.8, 1.0]$ to each of the originally stated assertions.

We added a set of axioms which allow to detect that some combinations of birth, death and publishing dates are inconsistent. For instance, the following axiom expresses the fact that a person who is younger than 15 years of age cannot publish something.

$$FamousResearcher \equiv \exists influenced.(\exists influenced.\top) \quad 0.6$$
$$>_{1850}(born) \sqcap \exists authorOf.<_{1865}(published) \sqsubseteq \bot \quad 0.82$$
$$\exists hasStudent.(FamousResearcher \sqcap >_{1830}(born) \sqcap \exists authorOf.<_{1910}(published)) \sqsubseteq EliteUniversity_{1850-1900} \quad 0.75$$

We added similar axioms to express, e.g., that nobody can be born after he died. All these axioms use only unary predicates. Thus we had to generate the axioms for different time points. We did this with a step size of 25, i.e., for each type of axiom we generated a set of axioms to cover the time span from 1700 to 2000.

The MAP state for this input data will be a subset of the weighted input assertions, which might contain incorrect and correct assertions. The MAP state will also contain the instantiations of all $EliteUniversity$ concepts. We compare the outcome of our approach against a gold standard, by assuming that each originally stated fact in DBpedia is correct, and each added fact is incorrect. Furthermore, we treat the $EliteUniversity$ query result of the first experiment as a gold standard to measure in how far the results are negatively affected by the noisy setting.

Table 2. Precision (P) and recall (R) scores for computing the MAP state with an increasing number of injected erroneous assertions.

Injection		0%	20%	40%	60%	80%	100%
Query	P	1.00	0.92	0.81	0.76	0.66	0.61
	R	1.00	0.93	0.86	0.79	0.76	0.70
Repair	P	-	0.83	0.83	0.83	0.83	0.83
	R	-	0.62	0.62	0.61	0.61	0.60
Time (s)		390	631	1313	2351	3336	4599

The results of our experiments are shown in Table 2. Since we randomly assigned weights, we repeated each experiment 10 times and present average scores. We were able to compute meaningful results in highly inconsistent settings. Even in a setting where we added 100% incorrect assertions (as much correct assertions as incorrect assertions), we are still able to achieve a query precision of 61% and a recall of 70%. The good query results are caused by the relatively good results for the repair, where we computed the fraction of incorrect statements that have been removed (recall of repair) and the fraction of correct removal decisions (precision of repair). Both values are stable for different injection rates. The measured runtimes do not increase linearly with respect to the size of the input data. This is in line with our expectation, since the original data (0% injection) set will only require a materialization, while each added incorrect assertion might be involved in a conflict resulting finally in a non trivial optimization problem.

6.2 Debugging LogEL(⊤) KBs

The objective of this experiment is debugging temporal LogEL(⊤) KBs. We define a set of domain specific temporal rules (similar to those in the example of Section 4.2) based on a common sense understanding of this dataset. Typical examples of such rules are the following ones.

$$(3)\ \forall x, t1, t2 : person(x) \wedge birthDate(x, t1) \wedge deathDate(x, t2) \Rightarrow <(t1, t2) \quad 0.7$$
$$(4)\ \forall x, y, t1, t2 : almaMater(x, y) \wedge established(y, t1) \wedge deathDate(x, t2) \Rightarrow <(t1, t2) \quad 0.5$$

Besides, we injected several types of erroneous facts to the extracted dataset. The results of our experiments are depicted in Table 3. The first column informs about the precision of the generated dataset. We have injected a fraction of 1%, 10%, 25%, 50%, 75%, and finally 100% incorrect statements to the dataset. Thus, the precision of the generated dataset varies from 0.99 (260K facts) to 0.5

(520K facts). For example, the last row of the table refers to a dataset where every second fact is incorrect. The recall of this dataset is always 1.0, because we never removed a fact of the extracted dataset. For all datasets, we have computed the MAP state, which corresponds to the repaired dataset. In doing so we computed precision, recall and F-measure for the debugging process and for the final outcome. The precision of the debugging process refers to the fraction of removed axioms or assertions that were indeed incorrect; recall refers in this context to the fraction of incorrect axioms or assertions that have been removed. The precision and recall of the repaired dataset is computed by comparing it to the originally extracted dataset. The rightmost column (ΔF) informs about the gain in F-measure that we computed by comparing the F-measure of the input dataset with the F-measure of the repaired dataset.

Table 3. Precision (P), Recall (R) and F-measure (F) for debugging process and repaired dataset.

Input	Debugging			Repaired Dataset			ΔF
P	P	R	F	P	R	F	
0.99	0.80	0.63	0.70	1.00	1.00	1.00	0.002
0.91	0.80	0.64	0.71	0.97	0.98	0.97	0.022
0.80	0.81	0.65	0.72	0.92	0.96	0.94	0.050
0.67	0.82	0.65	0.73	0.84	0.93	0.88	0.084
0.57	0.83	0.65	0.73	0.78	0.90	0.83	0.106
0.50	0.84	0.65	0.73	0.72	0.87	0.79	0.119

The results show that we are able to improve the data quality of an erroneous dataset. This is depicted in the increase of F-measure in the ΔF column for all test cases. With respect to the test case where every second fact is incorrect, we are able to increase the precision by 0.22 (from 0.5 up to 0.72), while recall is only reduced to 0.87 (from 1.0). Most of the removed facts are indeed incorrect (a repair precision of > 0.8 means that at least for 4 of 5 removals are proper removals), while we are able to detect more than half of the incorrect facts (debugging recall of the repair is > 0.5).

7 Related Work

Since in MLNs all variables and features are discrete, Hybrid MLNs that allow for both discrete and continuous variables in addition to limited numerical constraints have been introduced [36]. While our approach does not support continuous variables, the numerical constraints that we consider here are more expressive and the ability to compute them outside the cutting planes algorithm, enables us for far more richer constraints and functions. In the context of probabilistic programming, supporting numerical constraints in probablisitic reasoning is also explored [5, 4, 3].

The study of extending DLs to handle uncertainty and vagueness has gained momentum recently. There have been several proposals to add probabilities to various DLs. Probabilistic DLs can be classified in several dimensions. One possible classification is on the reasoning mechanism used: Markov logic networks (MLNs) and Bayesian networks. There exist some studies that employ MLNs to extend various DLs. The study in [21] extends $\mathcal{EL}^{++}$ with probabilistic uncertainty based on the annotation of axioms using MLNs. The main focus of this work is ranking queries in descending order of probability of atomic inferences which is different from the objective of this paper. Another study in [27], presents a probabilistic extension of the DL $\mathcal{EL}^{++}$ without nominals and CDs in MLN in order to find the most probable coherent ontology. In doing so, they have developed a reasoner for probabilistic OWL-EL called ELOG [29]. In this study, we extend this work in order to deal with CDs in addition to nominals and instances. In databases, MLNs have been used to create a probabilistic datalog called Datalog+/-. It is an extension of datalog that allows to express ontological axioms by using rule-based constraints [10]. The probabilistic extension of Datalog+/- uses MLNs as the underlying probabilistic semantics. The focus of this work is on scalable threshold query answering which is different from that of this work.

Other literatures extend DLs with Bayesian networks. Some notable works include: an extension of $\mathcal{EL}$ with Bayesian networks called $\mathcal{BEL}$ is presented in [6]. They study the complexity of reasoning under $\mathcal{BEL}$ to show that reasoning is intractable. However, their work does not discuss probabilities in the ABox and concrete domains are excluded. On the other hand, in [7], they added uncertainty to DL-Lite based on Bayesian networks. Additionally, they have shown that satisfiability test and query answering in probabilistic DL-Lite can be reduced to satisfiability test and query answering in the DL-Lite family. Further, it is proved that satisfiability checking and union of conjunctive query answering can be done in LogSpace in the data complexity. Query answering in probabilistic OWL QL, where ABox assertions are probabilistic and TBox axioms are deterministic, has been studied in [13]. They prove that only very simple conjunctive queries can be answered in PTime, while most queries are #P-hard.

It is possible to extend DLs with uncertainty by using Halpern's probabilistic first order logic [12]. These studies include Prob-$\mathcal{ALC}$ [25], P-$\mathcal{SHOIN}$(**D**) [35]. On the other hand, P-$\mathcal{SHIQ}$(**D**) uses probabilistic lexicographic entailment from probabilisitc default reasoning [19]. P-$\mathcal{SHOIN}$(**D**) and P-$\mathcal{SHIQ}$(**D**) support datatype reasoning in a probabilistic setting. While these logics are different from the one studied in this paper, they also do not support expressive CDs. In addition, while LogEL is based on log-linear models, P-$\mathcal{SHIQ}$(**D**) is based on Nilsson's probabilistic logic [28] and P-$\mathcal{SHOIN}$(**D**) is based on Halpern's probabilisitc FOL; both of these formalisms are different from log-linear models. A survey of probabilistic extensions of DLs can be found in [20].

Consequently, as discussed above, most of the studies that involve extending DLs to deal with uncertainty by using either Bayesian or MLNs often excluded CDs. This is partly due to either the lack of supporting features or the difficulty in dealing with them. In this paper, we studied a novel way of dealing with uncertainty involving CDs by log-linear models extended with numerical constraints.

8 Conclusion

In this paper, we extended MLN with numerical constraints. To show its usefulness, we have applied this framework to extend LogEL with instances and CDs. This work can be seen as a first step towards more practical uses of log-linear models with numerical constraints. We have illustrated the practical merits of the approach using an example application. What needs to be done is a systematic evaluation of the approach. Furthermore, there are two directions for improving the formalism. On the one hand, we can further extend the modeling capabilities by including other kinds of datatypes and probability distributions over attribute values. On the other hand, we will investigate tractable subsets of extended MLNs to enable polynomial time reasoning.

REFERENCES

[1] James F Allen, 'Maintaining knowledge about temporal intervals', *Communications of the ACM*, **26**(11), 832–843, (1983).

[2] Franz Baader, Sebastian Brandt, and Carsten Lutz, 'Pushing the el envelope', in *IJCAI*, volume 5, pp. 364–369, (2005).

[3] Vaishak Belle, Andrea Passerini, and Guy Van den Broeck, 'Probabilistic inference in hybrid domains by weighted model integration', in *Proceedings of 24th International Joint Conference on Artificial Intelligence (IJCAI)*, (2015).

[4] Vaishak Belle, Guy Van den Broeck, and Andrea Passerini, 'Hashing-based approximate probabilistic inference in hybrid domains', in *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence (UAI)*, (2015).

[5] Rodrigo de Salvo Braz, Ciaran O'Reilly, Vibhav Gogate, and Rina Dechter, 'Probabilistic inference modulo theories', (2016). to appear.

[6] Ismail Ilkan Ceylan and Rafael Penaloza, 'Bayesian description logics', in *Proceedings of the 27th International Workshop on Description Logics (DL 2014). CEUR Workshop Proceedings*, volume 1193, pp. 447–458, (2014).

[7] Claudia d'Amato, Nicola Fanizzi, and Thomas Lukasiewicz, 'Tractable reasoning with bayesian description logics', in *Scalable Uncertainty Management*, 146–159, Springer, (2008).

[8] M Despoina, Y Kazakov, and I Horrocks, 'Tractable extensions of the description logic el with numerical datatypes', *Journal of Automated Reasoning*, (2011).

[9] Lise Getoor, *Introduction to statistical relational learning*, MIT press, 2007.

[10] Georg Gottlob, Thomas Lukasiewicz, Maria Vanina Martinez, and Gerardo I Simari, 'Query answering under probabilistic uncertainty in datalog+/- ontologies', *Annals of Mathematics and Artificial Intelligence*, **69**(1), 37–72, (2013).

[11] Víctor Gutiérrez-Basulto, Jean Christoph Jung, Carsten Lutz, and Lutz Schröder, 'A closer look at the probabilistic description logic prob-el.', in *AAAI*, (2011).

[12] Joseph Y Halpern, 'An analysis of first-order logics of probability', *Artificial intelligence*, **46**(3), 311–350, (1990).

[13] Jean Christoph Jung and Carsten Lutz, 'Ontology-based access to probabilistic data with owl ql', in *The Semantic Web–ISWC 2012*, 182–197, Springer, (2012).

[14] Yevgeny Kazakov, Markus Krötzsch, and František Simančík, 'Practical reasoning with nominals in the $\mathcal{EL}$ family of description logics', in *Proceedings of the 13th International Conference on Principles of Knowledge Representation and Reasoning (KR'12)*, eds., Gerhard Brewka, Thomas Eiter, and Sheila A. McIlraith, pp. 264–274. AAAI Press, (2012).

[15] Yevgeny Kazakov, Markus Krötzsch, and František Simančík, 'The incredible ELK: From polynomial procedures to efficient reasoning with $\mathcal{EL}$ ontologies', *Journal of Automated Reasoning*, **53**, 1–61, (2013).

[16] Pavel Klinov, 'Pronto: a non-monotonic probabilistic description logic reasoner', in *Proceedings of the 5th European semantic web conference on The semantic web: research and applications*, pp. 822–826. Springer-Verlag, (2008).

[17] Daphne Koller and Nir Friedman, *Probabilistic graphical models: principles and techniques*, MIT press, 2009.

[18] Markus Krötzsch, 'Efficient rule-based inferencing for owl el', in *IJCAI*, volume 11, pp. 2668–2673, (2011).

[19] Thomas Lukasiewicz, 'Expressive probabilistic description logics', *Artificial Intelligence*, **172**(6), 852–883, (2008).

[20] Thomas Lukasiewicz, 'Expressive probabilistic description logics', *Artificial Intelligence*, **172**(6), 852–883, (2008).

[21] Thomas Lukasiewicz, Maria Vanina Martinez, Giorgio Orsi, and Gerardo I. Simari, 'Heuristic ranking in tightly coupled probabilistic description logics', in *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence, Catalina Island, CA, USA, August 14-18, 2012*, pp. 554–563, (2012).

[22] Carsten Lutz, 'Description logics with concrete domains-a survey', in *Advances in Modal Logic 4, papers from the fourth conference on "Advances in Modal logic," held in Toulouse (France) in October 2002*, pp. 265–296, (2002).

[23] Carsten Lutz, 'Combining interval-based temporal reasoning with general TBoxes', *Artificial Intelligence*, **152**(2), 235–274, (2004).

[24] Carsten Lutz and Maja Miličić, 'A tableau algorithm for description logics with concrete domains and general tboxes', *Journal of Automated Reasoning*, **38**(1-3), 227–259, (2007).

[25] Carsten Lutz and Lutz Schröder, 'Probabilistic description logics for subjective uncertainty.', in *KR*, (2010).

[26] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling, 'Never-ending learning', in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI-15)*, (2015).

[27] Mathias Niepert, Jan Noessner, and Heiner Stuckenschmidt, 'Log-linear description logics', in *IJCAI*, pp. 2153–2158, (2011).

[28] Nils J Nilsson, 'Probabilistic logic', *Artificial intelligence*, **28**(1), 71–87, (1986).

[29] Jan Noessner and Mathias Niepert, 'Elog: a probabilistic reasoner for owl el', in *Web Reasoning and Rule Systems*, 281–286, Springer, (2011).

[30] Jan Noessner, Mathias Niepert, and Heiner Stuckenschmidt, 'Rockit: Exploiting parallelism and symmetry for map inference in statistical relational models', in *Twenty-Seventh AAAI Conference on Artificial Intelligence*, (2013).

[31] Jan Noessner, Heiner Stuckenschmidt, Christian Meilicke, and Mathias Niepert, 'Completeness and optimality in ontology alignment debugging', in *Proceedings of the 9th International Workshop on Ontology Matching collocated with the 13th International Semantic Web Conference (ISWC 2014)*, pp. 25–36, Riva del Garda, Trentino, Italy, (2014).

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

[33] S Riedel, 'Improving the accuracy and efficiency of map inference for markov logic', in *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence, UAI 2008*, pp. 468–475, (2008).

[34] Sebastian Riedel, 'Improving the accuracy and efficiency of map inference for markov logic', *arXiv preprint arXiv:1206.3282*, (2012).

[35] Fabrizio Riguzzi, Elena Bellodi, Evelina Lamma, and Riccardo Zese, 'Reasoning with probabilistic ontologies', in *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pp. 4310–4316. AAAI Press, (2015).

[36] Jue Wang and Pedro M Domingos, 'Hybrid markov logic networks.', in *AAAI*, volume 8, pp. 1106–1111, (2008).