Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

A Declarative Approach to Data-Driven Fact Checking

Julien Leblay
Artificial Intelligence Research Center, AIST, Japan
firstname.lastname@aist.go.jp

Abstract

Fact checking is an essential part of any investigative work. For linguistic, psychological and social reasons, it is an inherently human task. Yet, modern media make it increasingly difficult for experts to keep up with the pace at which information is produced. Hence, we believe there is value in tools to assist them in this process. Much of the effort on Web data research has been focused on coping with incompleteness and uncertainty. Comparatively, dealing with context has received less attention, although it is crucial in judging the validity of a claim. For instance, what holds true in a US state, might not in its neighbors, e.g., due to obsolete or superseded laws.

In this work, we address the problem of checking the validity of claims in multiple contexts. We define a language to represent and query facts across different dimensions. The approach is non-intrusive and allows relatively easy modeling, while capturing incompleteness and uncertainty. We describe the syntax and semantics of the language. We present algorithms to demonstrate its feasibility, and we illustrate its usefulness through examples.

Introduction

Fact checking is the task of assessing the validity of a claim based on trusted sources. It is a basic component of journalism and to a larger extent any investigative task. The scale of information available on the Web, its incompleteness, its inconsistencies and the speed with which it spreads, have recently brought fact checking to the forefront in the media. Beyond "technical" limitations, there are external factors like culture or belief systems, that will likely prevent automation for a long time. Yet, as already advocated in the past (McCarthy 1993; Bienvenu, Deutch, and Suchanek 2012), contextual information can play a crucial role in interpreting the data. For instance, one might ask "Is John Doe eurosceptic?" The answer does not just depend on the person's reputation, but also on what "eurosceptic" means to different people, or the sources of information leading to a conclusion.

A number of probabilistic reasoning tools have been proposed over the years, but compiling uncertainty into a scalar value is often unsatisfactory. For instance, a knowledge base automatically extracted from the Web could contain the following facts, each of which inferred with enough confidence to be deemed trustworthy:

$$\mathsf{party(JohnDoe, Labour)}$$
$$\mathsf{party(JohnDoe, Tories)}$$

Several problems immediately appear in this example. Firstly, the knowledge base does not contain any explicit information about the person's position towards EU integration. To resolve the *incompleteness*, one might add more facts from other sources, in which case exploiting the provenance of those additional facts would be desirable. Another approach is to use axioms to check if the question is entailed by those facts. The following rule states that anyone belonging to the Conservative Party is eurosceptic:

$$\sigma_1 : \forall \mathsf{x}\ (\mathsf{party(x, Tories)} \rightarrow \mathsf{Eurosceptic(x)})$$

However, this is *context dependent* as not everyone might agree, or the rule might not have always held in the past.

Secondly, there is also *uncertainty* in that conflicts exist within the information source; either fact above might have been more accurate than the other over different periods of time, but there is no way to distinguish which one holds today. If one has write access to the data, it is be possible to add time-related facts to the knowledge base. But this requires a clear understanding of the underlying ontology, and does not protect against further redundant or conflicting statements.

Objective and contributions. This simple example highlights several issues that are all contextual to some extent: here *time* and *provenance* affect the interpretation of the question and thus the answer. In this work, we tackle the problem of answering queries in some predefined contexts, and exploring the answers as they vary. We aim to answer questions like: "According to sources A and B, is Mr. Doe eurosceptic?", "According to which sources could he be considered eurosceptic in 2010?", or "In which context is he eurosceptic with a confidence above 50%?"

To address this problem, we revisit some prior works on data management in the presence of incompleteness and uncertainty, namely probabilistic Datalog$^{\pm}$ (Gottlob et al. 2013), and contextual knowledge. Our contributions are as follows: (i) we define the syntax and semantics of a language to model incomplete, uncertain knowledge in multiple contexts, (ii) we describe a query language to assess the validity

of claims in such contexts, (iii) we provide algorithms for query answering, study their complexities and introduce optimizations for future implementation.

The next section recalls notions from the literature used in the remainder of the paper.

Preliminaries

We use a number of first-order logic (FOL) terms and notations, such as *constants* and *variables*, *atoms* and *formulas*, with which we assume familiarity. Unless we use conventional notations, capital letters denote sets, Greek lower case denotes formulas and functions, and Latin lower case denotes variables, constants or tuples.

Datalog$^{\pm}$

Datalog$^{\pm}$ (Cali et al. 2010) is a family of Datalog variants that were devised for efficient ontological querying. It has gained traction both as a theoretical and practical tool for the development of the Semantic Web and related problems, such as data integration, data exchange or query answering over incomplete data.

We assume a schema $\mathcal{R}$ as a set of relations of fixed arities, and the infinite sets Δ_C, Δ_N and $\mathcal{V}$, referring respectively to *constants*, *labeled nulls* and *variables*. While constants follow the unique name assumption, labeled nulls can be seen as *unknown constants*, and as such behave like variables (two distinct nulls may refer to the same value). A *term* is either a constant, a null or a variable. An *atom* of the form $\phi(\vec{t})$ is a relation symbol endowed with a tuple of terms. We may refer to *ground* atoms —whose terms belong to $\{\Delta_C \cup \Delta_N\}$— as *facts*. A database instance, or simply database, $\mathcal{D}$ is a set of facts abiding by $\mathcal{R}$.

Datalog$^{\pm}$ generalizes Datalog by allowing rules known as *tuple generating dependencies* (TGDs) of the form

$$\forall \vec{x}, \vec{y} \phi(\vec{x}, \vec{y}) \rightarrow \exists \vec{z} \psi(\vec{x}, \vec{z}) \quad (1)$$

where ϕ and ψ are conjunctions of atoms with terms in $\{\Delta_C \cup \mathcal{V}\}$, called the $body$ and the $head$ of the rule respectively. We often omit the quantifier for readability. The rule σ_1 given in the introduction is a TGD, so is the following:

Example 1. *All Eurosceptics support Brexit.*

$$\sigma_2 : \quad \textit{Eurosceptic(x)} \rightarrow \textit{supports(x, Brexit)}$$

A *conjunctive query* (CQ) is a rule of the form

$$Q(\vec{x}) \leftarrow \exists \vec{y} \phi(\vec{x}, \vec{y}) \quad (2)$$

where ϕ is a conjunction of atoms with terms in $\{\Delta_C \cup \mathcal{V}\}$. A CQ is *boolean* (BCQ), when $\vec{x}$ is empty, and *atomic* if its body consists of a single atom.

The Chase. The *chase* (Abiteboul, Hull, and Vianu 1995) is a procedure originally introduced to check query containment, now used in many database problems. It is a forward-chaining algorithm, proceeding in steps, starting from a database $\mathcal{D}_0$ and dependencies Σ. At *chase step* i, a TGD $\sigma \in \Sigma$ is selected and all homomorphisms μ from $body(\sigma)$ to $\mathcal{D}_{i-1}$ are found. For each μ, an extension μ' is obtained by adding mappings from each existential head variable to a fresh labeled null[1]. The chase step outputs a new database $\mathcal{D}_i = \mathcal{D}_{i-1} \cup \mathcal{D}'$, where $\mathcal{D}'$ contains all the facts obtained through applications of $\mu'(head(\sigma))$. Chase steps are applied exhaustively until a fixpoint is reached.

The output of the chase, called the *universal model* and denoted $chase(\mathcal{D}, \Sigma)$, is a data instance in which all the TGDs hold. The universal model may not be finite, and checking whether a set of dependencies has a finite model is undecidable, even when the database instance is fixed (Deutsch, Nash, and Remmel 2008). In spite of this, some classes of constraints enjoying terminating chase have been identified over the years (Fagin et al. 2003; Meier, Schmidt, and Lausen 2009). Guarded Datalog$^{\pm}$ requires all TGDs to be guarded, i.e., have a body atom containing all variables in the body. This ensures query answering has polynomial data-complexity even when the chase does not terminate. In this work, we assume TGDs belong to one of those classes.

Example 2. *Running the Chase with dependencies σ_1, σ_2 on the original two facts would add the following to the resulting instance.*

Eurosceptic(JohnDoe)
supports(JohnDoe, Brexit)

Datalog$^{\pm}$ also allows *equality-generating dependencies* (EGDs) of the form $\forall \vec{x}, \vec{y} \phi(\vec{x}, \vec{y}) \rightarrow x_i = x_j$, and negative constraints (NCs) such as $\forall \vec{x} \phi(\vec{x}) \rightarrow \bot$. These three types of dependencies capture a broad class of constraints, including (but not restricted to) primary keys (EGDs), foreign keys (TGDs), and other consistency checks, such as disjointness, with NCs. In this work, we ignore EGDs for simplicity.

Example 3. *The following NC (σ_3) states that one cannot support and oppose the same thing.*

$$\sigma_3 : \ \textit{opposes(x, y), supports(x, y)} \rightarrow \bot$$

A Datalog$^{\pm}$ ontology is a pair $(\mathcal{D}, \Sigma)$, where $\mathcal{D}$ is a finite database instance, and Σ is a set of TGDs and NCs.

Query answering. Let $(\mathcal{D}, \Sigma)$ be a Datalog$^{\pm}$ ontology. The answer of Q over $\mathcal{D} \cup \Sigma$, denoted $ans(Q, \mathcal{D}, \Sigma)$, is the set of tuples t taking values in $\{\Delta_C \cup \Delta_N\}$ such that there is a homomorphism $\mu : var(\phi) \rightarrow \{\Delta_C \cup \Delta_N\}$ with $\mu(\phi(\vec{x}, \vec{y})) \subseteq chase(\mathcal{D}, \Sigma)$ and $\mu(\vec{x}) = t$. If the query is boolean, then the answer is *true* iff there exists such a homomorphism, in which case we can write $\mathcal{D} \cup \Sigma \models Q$.

Chasing with NCs can lead to a contradiction in which case the chase stops. A BCQ is trivially true if any NC is violated.

Markov Logic Networks

MLN (Richardson and Domingos 2006) is one of many attempts to marry logical and probabilistic frameworks to reason about the world under uncertainty based on Markov Networks (MN). In brief, a program M is a set of pairs (ϕ_i, w_i), where ϕ_i is a FOL formula and w_i is a positive real number (called a weight).

[1] We assume the fresh nulls as taken from Skolem functions.

Intuitively, higher weights account for stronger formulas; those whose groundings reflect more plausible statements in the real world. The weights only have importance relatively to one another. They need not be restricted to a specific range, e.g. endowing all formulas with infinite weights yields FOL.

Example 4.

(¬support(x, Brexit) ∨ Eurosceptic(x), 3.0)
(¬CollegeGrad(x)∨opposes(x, Brexit), 2.0)

The above sentences state that supporting Brexit implies being Eurosceptic with a weight of 3.0, and that people with a college degree oppose Brexit, with a weight of 2.0.

Given a finite domain Δ'_C over which constants range, a possible world is a subset of the Herbrand Base $\mathcal{H}$. A probability distribution over all possible worlds is given as follows. For x a possible world:

$$P(x) = Z^{-1}\exp\Big[\sum_i w_i n_i(x)\Big] \tag{3}$$

where i ranges over the weighted formulas ϕ_i, n_i is the number of groundings making ϕ_i true in x and Z^{-1} is a normalization constant. The MN induced by an MLN features one node per atom in $\mathcal{H}$ and edges reflect how terms co-occur in the formulas. The marginal probability of a fact is the sum of probabilities of the worlds it belongs to.

Syntax

As usual, we assume a relational schema $\mathcal{R}$, the sets Δ_C, Δ_N, $\mathcal{V}$, and a database $\mathcal{D}$. We can derive a finite set of constants from $\mathcal{D}$, the "active domain" $\Delta'_C \subseteq \Delta_C$ (Abiteboul, Hull, and Vianu 1995). Let $K_1, \ldots, K_n$ be an ordered set of finite lattices, with $\oplus_i$ and $\otimes_i$ as join and meet operators respectively, and order relations $\preceq_i$, where $1 \leq i \leq n$. We always assume unique upper and lower bounds, denoted by $\top_i$ and $\bot_i$, i.e., for lattices with no unique upper (resp. lower) bound, $\top_i$ (resp. $\bot_i$) is a synthetic element added to the domain, into which any pair of upper (resp. lower) bounds join (resp. meet). For readability, we abbreviate the set of lattices to its product $\dot{K} = K_1 \times \cdots \times K_n$. The *product order* is denoted by $\dot{\preceq}$, and the associated meet and join operators by $\dot{\oplus}$ and $\dot{\otimes}$ respectively. $\dot{\top}$ and $\dot{\bot}$ denote the synthetic upper and lower bounds of $\dot{K}$ defined as above. We sometimes refer to tuples of $\dot{K}$ as *contexts* thereafter. We say that a context is *valid* if it does not contain any lower bound as component value. We now define the notion of *annotated formula* which is central to our model.

Definition 1 (Annotated formula)**.** *An* annotated formula *is of the form ϕ^A, where ϕ is a formula, and $A = \{\vec{a}_1 \ldots \vec{a}_n\}$ is a set of valid tuples in $\dot{K}$.*

We talk about *annotated fact* if ϕ is a *ground* atom. Intuitively, an annotated formula only holds within certain contexts captured by A. There is a parallel between annotated formula and *probabilistic dependencies* in (Gottlob et al. 2013), where dependencies can be annotated with sets of ground atoms, used to identify possible worlds in which the dependencies hold.

Definition 2 (Scenario)**.** *Let $\mathcal{D}$ and Σ be defined as usual, M a set of weighted formulas (ϕ_i, w_i), and $\dot{K}$ a set of contexts. A* scenario *$\mathcal{S}$ is a tuple $\langle \mathcal{D}, \Sigma, M, \dot{K}, \alpha \rangle$, where $\alpha : \{\mathcal{D} \cup \Sigma\} \to \mathcal{P}(\dot{K})_{\not\perp}$ is an annotation function, assigning a set of $\dot{K}$-tuples to each fact in $\mathcal{D}$ and dependency in Σ. Here, $\mathcal{P}(\dot{K})_{\not\perp}$ refers to all possible sets of valid $\dot{K}$-tuples*

A scenario includes a data instance $\mathcal{D}$, while the dependencies Σ feature "hard" constraints, dealing with incompleteness, inconsistencies and other matters. For instance, if the data comes from multiple sources, Σ may include data-exchange type of rules, to "export" all the data into a unique target schema. The MLN M models "soft" constraints. We disallow context tuples featuring lower bounds (*invalid* contexts), since it would deem an annotation to be *undefined* on some dimension. Finally, α adds "scopes" to facts or dependencies, allowing the context to be treated as orthogonal to the data. We resort to a function provided as an input alongside $\mathcal{D}$ and Σ , to allow applying our approach to legacy data and ontologies, while avoiding tampering with them. For instance, Example 5 could come from Linked Data repositories, with provenance annotations corresponding to the URL(s) each fact can be found at, and time annotations obtained using techniques such as in (Hoffart et al. 2013).

Example 5.

Let $\textsc{Years} = \{-\infty, 2006, \ldots, 2016, +\infty\}$ *be a finite set, where $-\infty$ and $+\infty$ denote arbitrary years in the far past and future, and* Time *a time interval lattice defined as* $\{[a, b] \mid a \leq b,\ a, b \in \textsc{Years}\} \backslash \{[-\infty, -\infty] \cup [+\infty, +\infty]\}$. *The order relation is the inclusion and $[-\infty, +\infty]$ and $[\,]$ denote its upper and lower bounds. We also assume a set of data sources* $\textsc{Src} = \{A, B, C\}$. *Let $\mathcal{S}$ be the scenario $\langle \mathcal{D}, \Sigma, M, \textsc{Time} \times \mathcal{P}(\textsc{Src}), \alpha \rangle$ such that applying α on $\mathcal{D} \cup \Sigma$ yields the set of annotated formulas:* {

party(JohnDoe, Tories)$^{([2013,+\infty],\{C\})}$
party(JohnDoe, Labour)$^{([-\infty,2014],\{A\})}$
bachelorFrom(JohnDoe, Imperial)$^{([2010,+\infty],\{A,B\})}$
opposes(JohnDoe, Brexit)$^{([2012,+\infty],\{B,C\})}$
Eurosceptic(JohnDoe)$^{([-\infty,2011],\{B\})}$
bachelorFrom(x, y)→CollegeGrad(x)$^{([-\infty,+\infty],\{A,B\})}$
party(x, Tories)→Eurosceptic(x)$^{([2007,2013],\{A,C\})}$
party(x, y)∧opposes(y, z)→opposes(x, z)$^{([-\infty,+\infty],\{A,B\})}$
supports(x, y)∧opposes(x, y)→ $\bot^{([-\infty,+\infty],\{A,B,C\})}$

}, *and the weighted formulas M are the following* {

(¬CollegeGrad(x) ∨ opposes(x, Brexit), 6.0)
(¬party(x, Tories)∨support(x, Brexit), 3.0)

}.

The scenario states that John Doe belonged to both the Tory and Labour parties, respectively from 2013 according to C, and until 2014 according to A. From A and B, we see that John holds a Bachelor's degree from Imperial since 2010, and that irrespective to time, having a Bachelor implies being a college graduate, and people oppose the same things as their party. We also know from B and C that John opposes the Brexit since 2012, but B also reports he was eurosceptic until 2011. According to A and C, between 2007 and 2013 being member of the Tories implied being

eurosceptic. All sources agree that one cannot support and oppose the same things, regardless of time. Finally, weighted formulas indicate that being a college graduate implies opposing the Brexit, while to a lesser degree, being a member of the Tories implies supporting it.

Query language. We now introduce two types of queries, *scope and support queries*. Recall that our ultimate goal is fact checking and as such we need a way to assess the truthfulness of *claims* in context. In our case, claims will take the form of conjunctive queries. We start by defining queries of the form "According to some given sources, is Mr. Doe eurosceptic?" and "According to whom was he eurosceptic in 2010?", described in the introduction.

Definition 3 (Scope Query)**.** *A* scope *query is of the form Q:b, where Q is a conjunction of atoms, and b is an optional "restriction" expressed as a conjunction of bindings $\#_i{=}v_i$, where $\#i$ refers to one of the components of $\dot{K}$, and v_i is a value in the domain $dom(K_i)$. A boolean scope query does not contain any variable.*

We also want to answer "reversed" queries, like "In which context is Mr. Doe eurosceptic with confidence above 50%?".

Definition 4 (Support Query)**.** *A (boolean) support query is of the form Q/s, where Q is a (boolean) scope query and s is a real number in $[0, 1]$.*

Semantics

To describe the semantics, we first modify the chase procedure to account for contextual annotations. We assume as usual a scenario $\mathcal{S} = \langle \mathcal{D}, \Sigma, M, \dot{K}, \alpha \rangle$.

Contextual chase. We call the *contextual chase*, denoted $chase_{\dot{K}}(\mathcal{D}, \Sigma, \alpha)$, the closure of $\mathcal{D}$ under Σ, in the context of $\dot{K}$. Its output is a pair $(\mathcal{D}', \alpha')$, such that $\mathcal{D}'$ is defined as the output of the conventional chase, and $\alpha' : \{\mathcal{D}' \cup \Sigma\} \to \mathcal{P}(\dot{K})_{\not\perp}$ is a new annotation function. α and α' coincide for every element in Σ, but may differ for inputs from $\mathcal{D}'$. We describe inductively how α' is obtained. Before the first chase step, $\alpha_0 = \alpha$. At the i^{th} step, let $\sigma : \phi_1 \wedge \cdots \wedge \phi_n \to \psi_1 \wedge \cdots \wedge \psi_m$ be the TGD under consideration, μ a homomorphism from $body(\sigma)$ to $\mathcal{D}_{i-1}$, and μ' the extension of μ to $head(\sigma)$. α_i is created as follows:

(i) $\forall \psi_k \in head(\sigma)$, $\alpha_i(\mu'(\psi_k)) = \alpha_{i-1}(\mu'(\psi_k)) \cup A$, with

$$A{=}\{\alpha(\sigma)\dot{\otimes}a_1\dot{\otimes}\ldots\dot{\otimes}a_n | a_1{\in}\mu(\phi_1), \ldots, a_n{\in}\mu(\phi_n)\}$$

(ii) for every other input, $\alpha_i(x) = \alpha_{i-1}(x)$

In other words, a chase step propagates sets of annotations of the facts from which μ originates. For each entry in the product of these sets, the $\dot{\otimes}$-operator is used to produce a new annotation which in turn is added to the function output for each derived head atom. We note that, strictly speaking, α_{i-1} may not be defined on all facts of $\mathcal{D}_i$, so we assume the empty set is returned by default in those cases, i.e., $\forall x \in \mathcal{D}_i \backslash \mathcal{D}_{i-1}, \alpha_{i-1}(x) = \emptyset$.

Chase steps for negative constraints are handled similarly. However, we do not stop immediately after a contradiction is derived, but rather, keep a set of annotations over $\perp$. We claim that this departure from the convention does not affect the termination of the chase.

Proposition 1. *Let $\mathcal{D}$, Σ, $\dot{K}$, and α be defined as usual. Then $chase_{\dot{K}}(\mathcal{D}, \Sigma, \alpha)$ terminates iff $chase(\mathcal{D}, \Sigma)$ terminates.*

Proof. In the absence of NCs, the output database $\mathcal{D}'$ coincides for the chase and the contextual chase by definition. If NCs are present, there exists a chase steps ordering such that no NC is fired until all TGDs are fired exhaustively. Finally, the number of annotations on any fact is bounded by $|\dot{K}|$. □

We also note that at any chase step, the contextual chase does a polynomial amount of extra work compared to the conventional chase.

We now define the notion of *projection*, which restricts a database and a set of dependencies to a given context.

Definition 5 (Projection)**.** *Given an annotation function α defined as usual, a* projection *of α over some context tuple $w \in \dot{K}$, denoted α_w, is a restriction on the* range *of α such that any annotation set A is replaced with $\{v \mid u \in A, v = u\dot{\otimes}w\}$.*

Intuitively, a projection narrows the scope of annotation on fact and dependencies to contexts *dominated* by the projection tuple. Tuples in $\dot{K}$ capture possible worlds, i.e., for some projection context $w \in \dot{K}$, a fact or dependency is considered to hold if it has some annotation tuple v such that $v\dot{\otimes}w$ is valid. This is a significant departure from MLNs and derived frameworks, where a possible world is a subset of the Herbrand base. Let ϕ be a ground formula in negation normal form, with atoms in $chase(\mathcal{D}, \Sigma)$, and $w \in \dot{K}$ some context tuple. We say that a scenario $\langle \mathcal{D}, \Sigma, M, \dot{K}, \alpha \rangle$ *contextually satisfies* ϕ w.r.t. w, denoted $\mathcal{D} \cup \Sigma \models_{\dot{K},\alpha_w} \phi$, if $chase_{\dot{K}}(\mathcal{D}, \Sigma, \alpha) = (\mathcal{D}', \alpha')$, and ϕ holds in $\mathcal{D}'$ under the projection α'_w.

Definition 6 (Contextual interpretation)**.**
Let $\mathcal{S} = \langle \mathcal{D}, \Sigma, M, \dot{K}, \alpha \rangle$ be a scenario. A contextual interpretation *is a probability distribution over contexts, such that $\forall x \in \dot{K}$,*

$$P(x) = Z^{-1}\exp\Big[\pi(x) \times \sum_i w_i n_i(x)\Big] \qquad (4)$$

where n_i is the number of groundings of a formula $\phi_i \in M$, such that $\mathcal{D} \cup \Sigma \models_{\dot{K},\alpha_x} \phi_i$, and $\pi(x)$ is a function returning a constant in $[0, 1]$ if $\mathcal{D} \cup \Sigma \models_{\dot{K},\alpha_x} \perp$, and 1 otherwise.

The penalty function π is an external parameter that modulates how strictly contradiction shall be treated. In our running example, we use a penalty of 0. This means that rather counter-intuitively, a context may have lower probability than another one it dominates, e.g. if it entails a contradiction while the other does not. For this reason, we make the

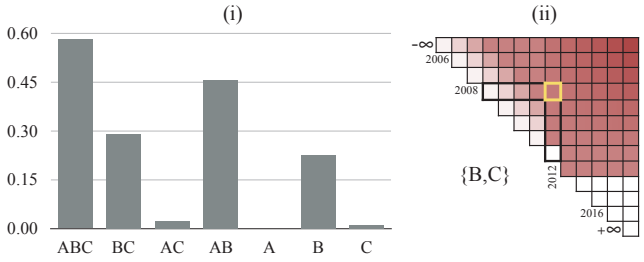

Figure 1: Contextual function for Eurosceptic(JohnDoe), with input bindings TIME=$[-\infty, +\infty]$ (i) and SRC={B,C} (ii). Cells in (ii) denote intervals, e.g. $[2008, 2012]$ in gold.

assumption that contexts are independent, despite the fact that projecting on two distinct contexts may yield the same valid fact and dependencies. Note that when the ontology is context-independent (i.e., α returns $\dot{\top}$ on any input), all contexts are equiprobable.

Definition 7 (Contextual function)**.** *A* contextual function *of a ground formula ϕ, is a function $\beta{:}\dot{K} \rightarrow [0, 1]$, defined as:*

$$\beta(x) = \sum_{\{w | w \dot{\preceq} x, \mathcal{D} \cup \Sigma \models_{\dot{K}, \alpha_w} \phi\}} P(w) \qquad (5)$$

The contextual function allows exploring how conclusions vary with contexts. To "drill down", it suffices to restrict the domain of β by binding parts of the inputs. For instance, let K_i be some arbitrary context with $\dot{K}$, with $1 \leq \mathrm{i} \leq \mathrm{n}$ and $\mathrm{v} \in \mathrm{dom}(\mathrm{K_i})$, then $\beta\,|_{\#\mathrm{i}=\mathrm{v}}\,(\mathrm{x})$ ranges over the tuple of $\dot{K}$ whose i^{th} position equals v.

Example 6. *Figure 1 visually represents the contextual function for the fact Eurosceptic(JohnDoe) of our running example. Figure 1 (i) depicts the function with the restriction* TIME $= [-\infty, +\infty]$ *applied. In this case, the function ranges over the power set of sources and shows for instance that B is the single source with the strongest confidence that John Doe is eurosceptic over that period. In Figure 1 (ii), the restriction is* SRC $= \{B, C\}$*. Each cell in the figure depicts a time interval, with darker colors corresponding to higher values. Looking at each 1-year interval sequentially, we see that confidence was low until 2011, dropped to 0 in 2012, peaked in 2013, and finally dropped to 0 again until the end.*

We now define query answering for scope queries which relies on contextual functions.

Definition 8 (Scope query answering)**.** *Let $Q{:}b$ be a scope query, $anScope(Q{:}b, \mathcal{S})$ is a set of results the form $\phi{:}\beta$, where ϕ is a conjunction of facts, and β is defined as in Equation 5 for ϕ, and there exist a homomorphism $\mu{:}vars(Q) \rightarrow \Delta_C \cup \Delta_N$, s.t. $\phi = \mu(Q)$ and $\mu(Q) \subseteq chase(\mathcal{D}, \Sigma)$. The restriction b translates into the corresponding restriction on β.*

Example 7. *The query "Who was eurosceptic between 2008 and 2014?" is expressed as Eurosceptic(X):* TIME $= [2008, 2014]$.
The answer is Eurosceptic(JohnDoe): $\{\{\mathrm{A}\} \rightarrow .028, \{B\} \rightarrow .053, \{C\} \rightarrow .042, \{A, B\} \rightarrow .135, \{B, C\} \rightarrow .138, \{A, C\} \rightarrow .113, \{A, B, C\} \rightarrow .304\}$.

In other words, taking all sources into account, the claims has a confidence of .304, while considering source A alone, the measure drops to .028.

Definition 9 (Support query answering)**.** *Let Q/s be a support query, $anSupport(Q/s, \mathcal{S})$ is the set of results of the form $\phi{:}E$, where E is a set of contexts such that $\forall e \in E, s \leq \beta(\mu(Q))$, with ϕ, μ and β defined as in Definition 8.*

Example 8. *The query Eurosceptic(x)/.5 asks in what context(s) the conjunction has at least 50% confidence. The answer is Eurosceptic(JohnDoe):* $\{\{[-\infty, 2014], \{A, B, C\}\}, \{[2006, 2015], \{A, B, C\}\}, \{[2008, 2016], \{A, B, C\}\}, \{[2009, +\infty], \{A, B, C\}\}, \dots\}$.

Algorithms

We now to turn to how implement query answering. We first describe a naïve algorithm for answering scope queries, inspired from the Threshold Algorithm in (Gottlob et al. 2013). The inputs are a scenario $\mathcal{S}$, a scope query $Q{:}b$. The algorithm loops over all contexts $w \in \dot{K}$. For each context, we project the data instance and dependencies over w and chases. We compute the individual score for w (Equation 4), and update Z. For each match r of Q in the chase closure, we update the result by iterating of over all contexts satisfying b, and dominated by w. For support queries, one needs to answer its scope query first, and keep only those context mapping to values exceeding the support.

Theorem 1. *Query answering is PTIME-complete in data complexity for both scope and support atomic queries, under guarded TGDs.*

Proof (Sketch). This follows from Theorem 1 in (Gottlob et al. 2013), and the observation that $|\dot{K}|$ is also considered constant here. Computing the chase under guarded TGDs for the query Q is PTIME-complete. This also holds for the contextual chase since, it only requires a polynomial amount of additional work. Updating the result requires an additional nested loop over $\dot{K}$, selecting the relevant context to the current function input, re-chasing and recompute the score. □

In general, one can expect $\dot{K}$ to be large, rendering the above algorithm of little practical use. It is common however to materialize the output of the Chase before evaluating a query over it. We can thus move the Chase outside the loop, using the observation that chasing once with α and projecting later on w has the same effect as projecting first and chasing afterwards. This also implies that not all context tuples need to be kept alongside each derived fact, but only the maximal ones, saving both space and time in the contextual chase. Finally, assuming the context is topologically sorted, it is possible to exploit results from prior rounds in the nested loop, and interrupt it early. This yields Algorithm 1. The function computeScore uses Equation 4 to compute the final probability of the context associated with w (for which we keep Z up-to-date in line 5). For cases where $\dot{K}$ is still too large, we plan to adapt the algorithm to existing approximate methods to estimate the most probable worlds.

input : $\mathcal{S} = \langle \mathcal{D}, \Sigma, M, \dot{K}, \alpha \rangle$ and $Q{:}b$
output: R, a set of results of the form $r{:}\beta$

$(\mathcal{D}', \alpha') \leftarrow chase_{\dot{K}}(\mathcal{D}, \Sigma, \alpha)$
foreach $w \in \dot{K}$ *in topological order* **do**
  $\alpha'_w \leftarrow$ projection of α' over w
  $p_w \leftarrow$computeScore$(M, \mathcal{D}', \alpha'_w)$
  $Z \leftarrow Z + p_w$
  if w *agrees with* b **then**
    foreach *match* r *of* Q **do**
      // Obtain $r{:}\beta$ from R if present
      if r *holds under* α'_w **then**
        $r{:}\beta(w) \leftarrow r{:}\beta(w) + p_w$
        foreach $v \dot{\preceq} w$ **do**
          $r{:}\beta(w) \leftarrow r{:}\beta(w) + r{:}\beta(v)$
        end
      end
      $R \leftarrow R \cup r{:}\beta$
    end
  end
end
return R

Algorithm 1: Semi-Naïve Scope Query Evaluation

Related Work

The current work draws connections between recurring problems in Web data: coping with uncertainty, incompleteness, inconsistencies and provenance. Many approaches have been devised to marry programming language or logic with probabilistic models (Goodman 2013), among which Relational Markov Network (Bunescu and Mooney 2004) or the BLOG language (Milch, Marthi, and Russell 2004), based on MN and Bayesian Networks (BN), respectively. Probabilistic databases, such as MYSTIQ (Boulos et al. 2005) and MayBMS (Huang et al. 2009), have also thrown a bridge between the relational and probabilistic models with the goal of scaling to large data instances. Dealing with incompleteness and uncertainly has also been explored in probabilistic Datalog (Fuhr 1995). The closest work to ours is probabilistic Datalog$^{\pm}$ (Gottlob et al. 2013). While our semantics and the problem we tackle are different, our syntax is inspired from it. However, in that setting dependencies are annotated with subsets of the Herbrand base to specify in which possible world they may hold. In this sense, we believe our syntax is more usable in practice. Other semantics for probabilistic Datalog$^{\pm}$ have been proposed (Riguzzi, Bellodi, and Lamma 2012). Provenance semirings can be used to jointly deal with incompleteness and uncertainty (Green 2009).

A large body of research has explored the use of context in logics for at least 20 years (McCarthy 1993), recent examples of which include (Joseph et al. 2016) and (Bozzato and Serafini 2014). Among others, in Multi-Context Logics (MCL) and Distributed Description Logics (DDL), contexts amount to local theories and so-called bridge rules (essentially TGDs) are used to exchange knowledge across them in a fixpoint computation. The notion of context in Contextual Knowledge Repositories (CKR) (Serafini and Homola 2012) is also close to ours. To the best of our knowledge, these do not deal with uncertainty in the way introduced here.

Computational journalism is an emerging research field (Cohen, Hamilton, and Turner 2011), of which computational fact checking is an active branch pioneered with (Wu et al. 2014). The authors formulate claims as SQL query templates and explore the parameter space to reveal how conclusions change. Two other recent works (Lehmann et al. 2012; Ciampaglia et al. 2015) fall into the category of "fact validation". The input is an RDF triple whose *truthfulness* is assessed w.r.t. some distance is computed from background data either coming from knowledge bases or extracted from Web pages. Fact validation also gets attention from the NLP community where for instance "textual entailment" is a popular problem (Mineshima et al. 2015). In (Hassan, Li, and Tremayne 2015), the authors aim to automatically classify claims worthy of further verification.

Our objective is to assess the validity of claims *within one or more contexts*. In context-sensitive probabilistic query answering (Ngo and Haddawy 1997) a query Q and a set of evidence E are used to construct a BN and compute $\mathrm{P}(\mathrm{Q} \mid \mathrm{E})$. In this work, we use lattice-valued annotations to model contexts, without interfering with the data itself. More generally, such annotations have long been a popular tool to "overlay" meta-data over existing data. Generalized Annotated Logics (Kifer and Subrahmanian 1992) pioneered this idea originally to deal with inconsistencies in logic. Recently, similar ideas have been extended and adapted to the RDF/SPARQL (Dividino et al. 2009; Zimmermann et al. 2012). The latter builds upon provenance semirings, and defines a more actionable method to deal multiple annotation domains. They suggest combining domains into functions mapping from one domain into another, an approach bearing similarity with contextual functions.

Discussion and future work

We have introduced a model and language to support the task of checking claims against background data. Observing that the task is inherently context sensitive, our approach combines earlier works on data management in the presence of incompleteness and inconsistency, as well as annotations, making context a first-class citizen in the process. The language and the algorithms proposed in this paper could serve as a foundation for assisted fact checking system implementations. We plan to extend the language, to allow other types of aggregation in contextual function (e.g. average, max) and other types of reasoning (e.g. weight learning), and evaluate its usefulness on real-world data.

Beyond fact checking, we think of adapting and applying our approach to knowledge base construction and incident analysis. The former has recently accomplished impressive progress, due in part to careful and efficient MLN implementations. However, inferring context-dependent facts remains a difficult challenge. Likewise, our approach could be instrumental in building better systems for incident prevention and recovery, making it possible to explore possible causes of events from different angles.

Acknowledgments

This paper is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO). The author would like to thank M. Benedikt, S. Lynden, N. Schwind and P. Senellart as well as the reviewers for their useful comments.

References

Abiteboul, S.; Hull, R.; and Vianu, V. 1995. *Foundations of databases*. Addison-Wesley Longman Publishing Co., Inc.

Bienvenu, M.; Deutch, D.; and Suchanek, F. M. 2012. Provenance for Web 2.0 data. In *Workshop on Secure Data Management*, 148–155. Springer.

Boulos, J.; Dalvi, N.; Mandhani, B.; Mathur, S.; Re, C.; and Suciu, D. 2005. MYSTIQ: a system for finding more answers by using probabilities. In *Proceedings of the International Conference on Management of Data*, 891–893. ACM.

Bozzato, L., and Serafini, L. 2014. Combining reasoning on Semantic Web metadata. In *European Conference on Artificial Intelligence*, 979–980.

Bunescu, R., and Mooney, R. J. 2004. Relational Markov networks for collective information extraction. In *Workshop on Statistical Relational learning*.

Cali, A.; Gottlob, G.; Lukasiewicz, T.; Marnette, B.; and Pieris, A. 2010. Datalog+/-: A family of logical knowledge representation and query languages for new applications. In *Annual IEEE Symposium on Logic in Computer Science*, 228–242. IEEE.

Ciampaglia, G. L.; Shiralkar, P.; Rocha, L. M.; Bollen, J.; Menczer, F.; and Flammini, A. 2015. Computational fact checking from knowledge networks. *PloS one* 10(6):e0128193.

Cohen, S.; Hamilton, J. T.; and Turner, F. 2011. Computational journalism. *Commun. of the ACM* 54(10):66–71.

Deutsch, A.; Nash, A.; and Remmel, J. 2008. The chase revisited. In *Proceedings of the Symposium on Principles of Database Systems*, 149–158. ACM.

Dividino, R.; Sizov, S.; Staab, S.; and Schueler, B. 2009. Querying for provenance, trust, uncertainty and other meta knowledge in rdf. *Web Semantics: Science, Services and Agents on the WWW* 7(3).

Fagin, R.; Kolaitis, P. G.; Miller, R. J.; and Popa, L. 2003. Data Exchange: Semantics and query answering. In *International Conference on Database Theory*, 207–224.

Fuhr, N. 1995. Probabilistic Datalog: a logic for powerful retrieval methods. In *Proceedings of the Int'l Conference on Research and Development in Information Retrieval*, 282–290. ACM.

Goodman, N. D. 2013. The principles and practice of probabilistic programming. *ACM SIGPLAN Notices* 48(1):399–402.

Gottlob, G.; Lukasiewicz, T.; Martinez, M. V.; and Simari, G. I. 2013. Query answering under probabilistic uncertainty in Datalog+/- ontologies. *Annals of Mathematics and Artificial Intelligence* 69(1):37–72.

Green, T. J. 2009. Models for incomplete and probabilistic information. *Managing and Mining Uncertain Data* 9.

Hassan, N.; Li, C.; and Tremayne, M. 2015. Detecting check-worthy factual claims in presidential debates. In *Proceedings of International on Conference on Information and Knowledge Management*, 1835–1838. ACM.

Hoffart, J.; Suchanek, F. M.; Berberich, K.; and Weikum, G. 2013. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. In *International Joint Conference on Artificial Intelligence*, 3161–3165. AAAI Press.

Huang, J.; Antova, L.; Koch, C.; and Olteanu, D. 2009. MayBMS: a probabilistic database management system. In *Proceedings of the International Conference on Management of Data*, 1071–1074. ACM.

Joseph, M.; Kuper, G. M.; Mossakowski, T.; and Serafini, L. 2016. Query answering over contextualized RDF/OWL knowledge with forall-existential bridge rules: Decidable finite extension classes. *Semantic Web* 7(1):25–61.

Kifer, M., and Subrahmanian, V. 1992. Theory of generalized annotated logic programming and its applications. *The Journal of Logic Programming* 12(4):335–367.

Lehmann, J.; Gerber, D.; Morsey, M.; and Ngomo, A.-C. N. 2012. Defacto-deep fact validation. In *International Semantic Web Conference*, 312–327. Springer.

McCarthy, J. 1993. Notes on formalizing context. In *International Joint Conference on Artificial Intelligence*, 555–562.

Meier, M.; Schmidt, M.; and Lausen, G. 2009. On chase termination beyond stratification. *Proceedings of the VLDB Endowment* 2(1):970–981.

Milch, B.; Marthi, B.; and Russell, S. 2004. BLOG: Relational modeling with unknown objects. In *Workshop on Statistical Relational learning*, 67–73.

Mineshima, K.; Martínez-Gómez, P.; Miyao, Y.; and Bekki, D. 2015. Higher-order logical inference with compositional semantics. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2055–2061.

Ngo, L., and Haddawy, P. 1997. Answering queries from context-sensitive probabilistic knowledge bases. *Theoretical Computer Science* 171(1-2):147–177.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Machine learning* 62(1-2):107–136.

Riguzzi, F.; Bellodi, E.; and Lamma, E. 2012. Probabilistic Datalog+/- under the distribution semantics. In *International Workshop on Description Logics*, volume 846, 1613–0073.

Serafini, L., and Homola, M. 2012. Contextualized knowledge repositories for the Semantic Web. *Web Semantics: Science, Services and Agents on the WWW* 12-13:64–87.

Wu, Y.; Agarwal, P. K.; Li, C.; Yang, J.; and Yu, C. 2014. Toward computational fact-checking. *Proceedings of the VLDB Endowment* 7(7):589–600.

Zimmermann, A.; Lopes, N.; Polleres, A.; and Straccia, U. 2012. A general framework for representing, reasoning and querying with annotated Semantic Web data. *Web Semantics: Science, Services and Agents on the WWW* 11:72 – 95.