Inconsistency-Tolerant Reasoning in Datalog$^{\pm}$ Ontologies via an Argumentative Semantics

M. V. Martinez   C. A. D. Deagustini   M. A. Falappa   G. R. Simari

Artificial Intelligence Research and Development Laboratory (LIDIA)
Department of Computer Science and Engineering
Universidad Nacional del Sur

Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET)

Presentation Outline

Motivation

- Inconsistency management is an important problem in real world applications, *e.g.*, when combining knowledge for reasoning purposes provided by different users.
- Many methods and frameworks have been developed to deal with inconsistency in Artificial Intelligence, the Semantic Web and Database Theory communities, providing us with deep insight on the options available to define what constitutes an *answer to a query in the presence of inconsistencies*.
- Here, we argue that the process of conflict resolution could be carried out through logical reasoning, using as much information as possible to weigh out conflicting pieces of information.

Outline

- We introduce *Defeasible* Datalog$^{\pm}$ ontologies [8] extending classical Datalog$^{\pm}$ with defeasible atoms and defeasible *tuple-generating dependencies* (or TGDs) that allow consequences to represent statements whose acceptance can be challenged.
- Conflicts are resolved through an argumentative dialectical process that considers reasons for and against potential conclusions and decides which are the ones that can be obtained (*warranted*) from the knowledge base.

Outline

- We show that the set of consequences that our framework entails are guaranteed to satisfy a very important property (the *NCE property*): no conflicting atoms can be entailed from a Defeasible Datalog$^{\pm}$ ontology.
- Some of these consequences cannot be obtained through existing inconsistency-tolerant semantics.
- We establish the relationship between our framework and several of these semantics.

Knowledge Representation in Defeasible Datalog$^\pm$

- In Datalog$^\pm$, knowledge is represented by combining several different components:
  - Atoms (facts) in a database,
  - tuple-generating dependencies (TGDs), used as inference rules,
  - negative constraints (NCs), expressing conjunctions of atoms that cannot be true together.
- *Equality-generating dependencies (EGDs)* can also be added to Datalog$^\pm$ ontologies; here, we do not consider EGDs since for our purposes they can be modeled in other ways (see [4] for details).

Database

- An *atomic formula* (or *atom*) **a** has the form $P(t_1, \ldots, t_n)$, where P is an n-ary predicate, and $t_1, \ldots, t_n$ are data constants, nulls, or variables.
- A *database (instance)* F is a set of atoms.
- Databases are used to represent known facts about the world.

Example (Database)

$$\begin{aligned} F = \; & \{\mathit{collaborates}(\mathit{will}, \mathit{fbi}), \\ & \mathit{security\_agency}(\mathit{fbi}), \mathit{psychiatrist}(\mathit{hannibal}, \mathit{will})\} \end{aligned}$$

Here, for instance, $\mathit{collaborates}(\mathit{will}, \mathit{fbi})$ states that Will assists the FBI.

Tuple-Generating Dependencies (TGDs)

- New information can be inferred in Datalog$^{\pm}$ by means of *tuple-generating dependencies (TGDs)*.
- A TGD σ is a first-order formula $\forall \mathbf{X} \forall \mathbf{Y}\, \Phi(\mathbf{X}, \mathbf{Y}) \rightarrow \exists \mathbf{Z}\, \Psi(\mathbf{X}, \mathbf{Z})$, where $\Phi(\mathbf{X}, \mathbf{Y})$ and $\Psi(\mathbf{X}, \mathbf{Z})$ are conjunctions of atoms (without nulls).
- The application of a TGD generates a new (strict) atom in the form of the head of the TGD.

Example (TGDs)

$$psychiatrist(S, P) \rightarrow in\_therapy(P)$$
$$in\_therapy(P) \rightarrow \exists S\, psychiatrist(S, P)$$

Negative Constraints (NCs)

- *Negative constraints (NCs)* are used in Datalog$^{\pm}$ to express conflicts among atoms.
- A *negative constraint* γ is a first-order formula of the form $\forall \mathbf{X} \Phi(\mathbf{X}) \rightarrow \perp$, where $\Phi(\mathbf{X})$ is a conjunction of atoms (without nulls) over $\mathcal{R}$ (the relational schema).
- We will restrict our attention to binary denial constraints since, as we will show later, this class of constraints suffices for the formalization of the concept of conflicting atoms.

Example (NCs)

$$\mathit{risky\_job}(P) \wedge \mathit{unstable}(P) \rightarrow \perp$$

This NC states that an unstable person cannot have a job that involves some sort of danger.

Defeasible Atoms

- We use *defeasible atoms* to represent statements whose acceptance can be challenged.
- The database instance of a defeasible Datalog$^\pm$ ontology can be seen as having two parts, a set of facts (*i.e.*, strict knowledge), denoted with F, and a set of defeasible atoms, denoted with D.

Example (Defeasible atoms)

$$D = \{victim(abigail)\}$$

This statement says that Abigail is *presumably* a victim of a crime, it is expressed as a defeasible atom since some suspicious actions from her indicate she might be an accomplice of the perpetrator.

Defeasible TGDs

- We also add defeasibility to express connections between pieces of information that are of a weaker nature than TGDs; *defeasible TGDs* are rules of the form $\Upsilon(\mathbf{X}, \mathbf{Y}) \succ \exists \mathbf{Z}\, \Psi(\mathbf{X}, \mathbf{Z})$, where $\Upsilon(\mathbf{X}, \mathbf{Y})$ and $\Psi(\mathbf{X}, \mathbf{Z})$ are conjunctions of atoms.
- As in DeLP's defeasible rules [5], acceptance of the body of a defeasible rule does not always lead to the acceptance of the head, which means that consequences of such rule can be challenged.
- Thus, the application of a defeasible TGDs generates a new *defeasible* atom obtained from the head if it does not exists in $F \cup D$.

Example (Defeasible TGDs)

$$lives\_depend\_on(A) \wedge works\_in(A, P) \succ risky\_job(P)$$

Usually, if someone works in a place that protects life then that person has a perilous job.

Defeasible Ontologies

- A defeasible Datalog$^{\pm}$ ontology is formed by the presented components, *i.e.*, $KB = (F, D, \Sigma_T, \Sigma_D, \Sigma_{NC})$.

Example (Defeasible Datalog$^{\pm}$ ontology)

$$KB = \left\{ \begin{array}{rl} F: & \{collaborates(will, fbi), security\_agency(fbi), \\ & \ psychiatrist(hannibal, will)\} \\ D: & \{victim(abigail)\} \\ \Sigma_T: & \{collaborates(P, A) \rightarrow works\_in(A, P), \\ & \ psychiatrist(S, P) \rightarrow in\_therapy(P)\} \\ \Sigma_D: & \{in\_therapy(P) \succ unstable(P), \\ & \ lives\_depend\_on(A) \wedge works\_in(A, P) \succ risky\_job(P), \\ & \ security\_agency(A) \succ lives\_depend\_on(A)\} \\ \Sigma_{NC}: & \{risky\_job(P) \wedge unstable(P) \rightarrow \bot\} \end{array} \right\}$$

Derivations

- As in classical Datalog$^{\pm}$, derivations from a defeasible Datalog$^{\pm}$ ontology rely on the application of (strict or defeasible) TGDs.

Definition

Let $KB = (F, D, \Sigma_T, \Sigma_D, \Sigma_{NC})$ be a defeasible Datalog$^{\pm}$ ontology and L an atom. An *annotated derivation* ∂ of L from KB consists of a finite sequence $[R_1, R_2, \ldots, R_n]$ such that R_n is L, and each atom R_i is either: (i) R_i is a fact or defeasible atom, *i.e.*, $R_i \in F \cup D$, or (ii) there exists a TGD $\sigma \in \Sigma_T \cup \Sigma_D$ and a homomorphism h such that $h(head(\sigma)) = R_i$ and σ is applicable to the set of all atoms and defeasible atoms that appear before R_i in the sequence.

- When no defeasible atoms and no defeasible TGDs are used in a derivation, we say the derivation is a *strict derivation*, otherwise it is a *defeasible derivation*.

Derivations - Example

- We say that an atom a is strictly derived from KB iff there exists a strict derivation for a from KB, denoted with $KB \vdash a$, and a is defeasibly derived from KB iff there exists a defeasible derivation for a from KB and no strict derivation exists, denoted with $KB \mid\!\sim a$.

Example

From the previously presented defeasible Datalog$^{\pm}$ ontology we can get the following (minimal) annotated derivation for atom $unstable(will)$:

$$\begin{aligned} \partial \ = \ \big[\ & psychiatrist(hannibal, will), \\ & psychiatrist(S, P) \rightarrow in\_therapy(P), \\ & in\_therapy(will), \\ & in\_therapy(P) \succ unstable(P), \\ & unstable(will)\big] \end{aligned}$$

Then, we have $KB \vdash in\_therapy(will)$ and $KB \mid\!\sim unstable(will)$.

Relation between Classic and Defeasible Datalog$^\pm$

- It can be shown that classical query answering in Datalog$^\pm$ ontologies (denoted as $KB \vdash L$) is equivalent to query answering in defeasible Datalog$^\pm$ ontologies (denoted as $KB \mathrel{|\!\sim} L$).

Proposition

Let L be a ground atom, $KB = (F, D, \Sigma_T, \Sigma_D, \Sigma_{NC})$ be a defeasible Datalog$^\pm$ ontology, $KB' = (F \cup D, \Sigma'_T \cup \Sigma_{NC})$ is a classical Datalog$^\pm$ ontology where

$$\Sigma'_T = \Sigma_T \cup \{\Upsilon(\mathbf{X}, \mathbf{Y}) \rightarrow \exists \mathbf{Z}\, \Psi(\mathbf{X}, \mathbf{Z}) \mid \Upsilon(\mathbf{X}, \mathbf{Y}) \succ\!\!- \exists \mathbf{Z}\, \Psi(\mathbf{X}, \mathbf{Z})\}.$$

Then, $KB' \models L$ iff $KB \vdash L$ or $KB \mathrel{|\!\sim} L$.

- Therefore, the set of atoms derived from a defeasible Datalog$^\pm$ ontology is the same as the one that is entailed from the corresponding traditional Datalog$^\pm$ ontologies; what changes is the nature of those derivations, *i.e.*, some atoms might be defeasibly derived.

Relation between Classic and Defeasible Datalog$^{\pm}$

- As a direct consequence, all the existing work done for Datalog$^{\pm}$ directly applies to defeasible Datalog$^{\pm}$.
- Despite the equivalence, defeasible reasoning allows the possibility of managing conflicts in a more sensible way; *i.e.*, considering in the resolution process aspects of the nature of the different pieces of knowledge in conflict and/or the way they are derived from existing knowledge.
- To better exploit the defeasible characteristics of these ontologies, we develop an argumentation-based procedure to answer queries in defeasible Datalog$^{\pm}$ ontologies.

Conflicts in Defeasible Datalog$^{\pm}$

- Conflicts in defeasible Datalog$^{\pm}$ ontologies come, as in classical Datalog$^{\pm}$, from the violation of negative constraints.

Definition

Given a set of negative constraints Σ_{NC}, two ground atoms (possibly with nulls) a and b are said to be *in conflict* relative to Σ_{NC} iff there exists an homomorphism h such that $h(body(\upsilon)) = a \wedge b$ for some $\upsilon \in \Sigma_{NC}$.

- We say that a set of atoms is a *conflicting* set of atoms relative to Σ_{NC} if and only if there exist at least two atoms in the set that are in conflict relative to Σ_{NC}, otherwise will be called *non-conflicting*.

Conflicts in Defeasible Datalog$^{\pm}$ - Example

Example

$$KB = \left\{ \begin{array}{rl} F: & \{collaborates(will, fbi), security\_agency(fbi), \\ & psychiatrist(hannibal, will)\} \\ D: & \{victim(abigail)\} \\ \Sigma_T: & \{collaborates(P, A) \rightarrow works\_in(A, P), \\ & psychiatrist(S, P) \rightarrow in\_therapy(P)\} \\ \Sigma_D: & \{in\_therapy(P) \succ unstable(P), \\ & lives\_depend\_on(A) \wedge works\_in(A, P) \succ risky\_job(P), \\ & security\_agency(A) \succ lives\_depend\_on(A)\} \\ \Sigma_{NC}: & \{risky\_job(P) \wedge unstable(P) \rightarrow \bot\} \end{array} \right\}$$

The set of atoms $\{unstable(will), risky\_job(will)\}$ is a conflicting set relative to Σ_{NC}. However, this is not the case for the sets $\{collaborates(will, fbi), psychiatrist(hannibal, will), security\_agency(fbi)\}$ or $\{unstable(will), risky\_job(hannibal)\}$.

Arguments

- Whenever defeasible derivations of conflicting atoms exist, we use a dialectical process to decide which information prevails, *i.e.*, which piece of information is such that no acceptable reason can be put forward against it.
- These reasons are backed up by arguments, which are structures that support a claim from evidence through the use of a reasoning mechanism.
- Given a defeasible Datalog$^{\pm}$ ontology, an argument $\mathcal{A}$ for a claim L is a minimal (under $\subseteq$) set of facts, defeasible atoms, TGDs, and defeasible TGDs contained in KB, such that L is derived from it and no conflicting atoms can be derived from it.
- The set of all arguments that can be built from KB is denoted $\mathbb{A}_{KB}$.

Arguments

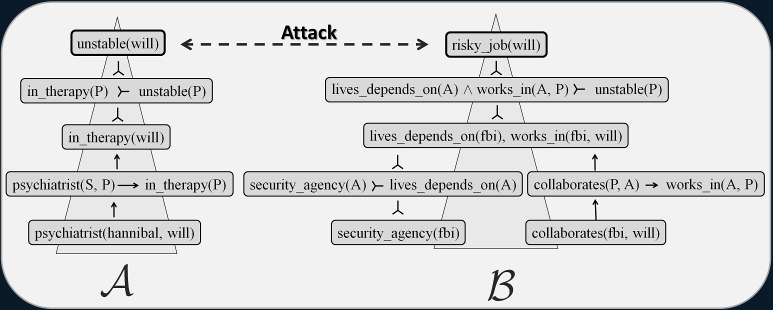
- Answers to atomic queries are supported by arguments built from the ontology.

- However, it is possible to build arguments for conflicting atoms, and in this situation arguments it is said that these arguments *attack* each other.

- An argument $\langle \mathcal{A}_1, L_1 \rangle$ counter-argues, rebuts, or attacks $\langle \mathcal{A}_2, L_2 \rangle$ at literal L, iff there exists a sub-argument $\langle \mathcal{A}, L \rangle$ of $\langle \mathcal{A}_2, L_2 \rangle$ such that L and L_1 conflict.

Example

$$KB = \left\{ \begin{array}{rl} F: & \{collaborates(will, fbi), security\_agency(fbi), \\ & psychiatrist(hannibal, will)\} \\ D: & \{victim(abigail)\} \\ \Sigma_T: & \{collaborates(P, A) \rightarrow works\_in(A, P), \\ & psychiatrist(S, P) \rightarrow in\_therapy(P)\} \\ \Sigma_D: & \{in\_therapy(P) \succ unstable(P), \\ & lives\_depend\_on(A) \wedge works\_in(A, P) \succ risky\_job(P), \\ & security\_agency(A) \succ lives\_depend\_on(A)\} \\ \Sigma_{NC}: & \{risky\_job(P) \wedge unstable(P) \rightarrow \bot\} \end{array} \right\}$$



Attack
unstable(will)
in_therapy(P) ≻ unstable(P)
in_therapy(will)
psychiatrist(S, P) ⟶ in_therapy(P)
psychiatrist(hannibal, will)
risky_job(will)
lives_depends_on(A) ∧ works_in(A, P) ≻ unstable(P)
lives_depends_on(fbi), works_in(fbi, will)
security_agency(A) ≻ lives_depends_on(A)
collaborates(P, A) → works_in(A, P)
security_agency(fbi)
collaborates(fbi, will)
𝒜
ℬ

Defeat and Argumentation Frameworks

- Once the attack relation is established between arguments, it is necessary to analyze whether the attack is strong enough so one of the arguments can *defeat* the other.
- Given an argument $\mathcal{A}$ and a counter-argument $\mathcal{B}$, a comparison criterion is used to determine if $\mathcal{B}$ is preferred to $\mathcal{A}$ and, therefore, *defeats* $\mathcal{A}$.
- Argumentation frameworks arise when we consider the conflict and defeat relations among arguments.

Definition

Given a defeasible Datalog$^{\pm}$ ontology KB defined over a relational schema $\mathcal{R}$, a *Datalog$^{\pm}$ argumentation framework* $\mathfrak{F}$ is a tuple $\langle \mathcal{L}_{\mathcal{R}}, \mathbb{A}_{KB}, \succ \rangle$, where $\succ$ specifies a preference relation defined over $\mathbb{A}_{KB}$.

Warrant

- To decide whether an argument $\langle \mathcal{A}_0, L_0 \rangle$ is undefeated within a Datalog$^\pm$ argumentation framework, all its defeaters must be considered, also there may be defeaters for those counter-arguments to be considered as well, which leads to dialectical structures called argumentation lines [5].
- The dialectical process considers all possible admissible argumentation lines for an argument, which together form a dialectical tree.
- Finally, argument evaluation is made over dialectical trees by means of a *marking* or *labelling* criterion, where a node in the tree is undefeated iff all of its children are defeated or it is a leaf.

Warrant and Entailment

- An atom L is *warranted* in $\mathfrak{F}$ (through a dialectical tree $\mathcal{T}$) iff there exists an argument $\mathcal{A}$ for L such that the root of $\mathcal{T}(\langle \mathcal{A}, L\rangle)$ is marked as undefeated.
- We say that L is entailed from KB (through $\mathfrak{F}$), denoted with $KB \models_{\mathfrak{F}} L$, iff it is *warranted* in $\mathfrak{F}$.

Example

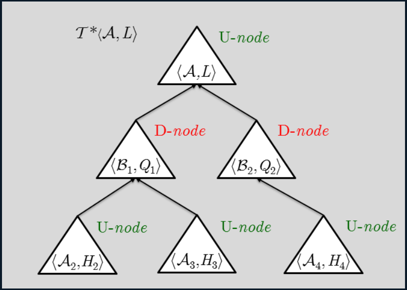
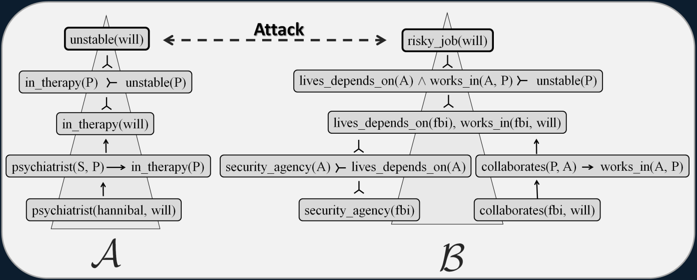

$\mathcal{T}^*\langle \mathcal{A}, L\rangle$
U-*node*
$\langle \mathcal{A}, L\rangle$
D-*node*
$\langle \mathcal{B}_1, Q_1\rangle$
D-*node*
$\langle \mathcal{B}_2, Q_2\rangle$
U-*node*
$\langle \mathcal{A}_2, H_2\rangle$
U-*node*
$\langle \mathcal{A}_3, H_3\rangle$
U-*node*
$\langle \mathcal{A}_4, H_4\rangle$

Figure: The warranting of atom L.

Warrant and Entailment



Attack
unstable(will)
in_therapy(P) ≻ unstable(P)
in_therapy(will)
psychiatrist(S, P) ⟶ in_therapy(P)
psychiatrist(hannibal, will)
$\mathcal{A}$
risky_job(will)
lives_depends_on(A) ∧ works_in(A, P) ≻ unstable(P)
lives_depends_on(fbi), works_in(fbi, will)
security_agency(A) ≻ lives_depends_on(A)
collaborates(P, A) → works_in(A, P)
security_agency(fbi)
collaborates(fbi, will)
$\mathcal{B}$

Example

Continuing with *KB*, consider its corresponding Datalog$^{\pm}$ argumentation framework $\mathfrak{F}$, and assume that $\succ$ in $\mathfrak{F}$ is such that $\mathcal{A} \succ \mathcal{B}$, and in that case $\mathcal{A}$ defeats $\mathcal{B}$.

In such case, the atom $unstable(will)$ is warranted through $\mathfrak{F}$, *i.e.*, $KB \models_{\mathfrak{F}} unstable(will)$.

The reason for this is that, under the given assumption, $unstable(will)$ is warranted through the dialectical tree that contains only the node $\langle \mathcal{A}, unstable(will) \rangle$ (its undefeated root node).

Coherence of Entailment in Defeasible Datalog$^{\pm}$

- The following proposition establishes that no conflicting sets of atoms can be entailed/warranted from a Datalog$^{\pm}$ argumentation framework.

Proposition

Let $KB = (F, D, \Sigma_T, \Sigma_D, \Sigma_{NC})$ be a defeasible Datalog$^{\pm}$ ontology. No two atoms L_1 and L_2 that are warranted in $\mathfrak{F}$ are conflicting relative to Σ_{NC}.

- This property follows from the internal structure of arguments, the definition of the conflict relation, and the way the dialectical analysis is defined.

Non-Conflicting Entailment (NCE)

- We call the presented property **Non-Conflicting Entailment (NCE)**.
- Given a knowledge base $\mathcal{K}$ and a set of binary negative constraints Σ_{NC}, any entailment operator $\models$ satisfies the NCE property *iff* for any two atoms L_1 and L_2 such that $\mathcal{K} \models L_1$ and $\mathcal{K} \models L_2$ it holds that they are non-conflicting relative to Σ_{NC}.
- Intuitively, this means that atoms entailed by $\mathcal{K}$ do not violate any negative constraint in Σ_{NC}. That is, for any two atoms L_1 and L_2 such that $\mathcal{K} \models L_1$ and $\mathcal{K} \models L_2$ it cannot happen that $L_1 \wedge L_2 \rightarrow \bot$.
- For instance, for the knowledge base KB in the previous example, depending on the preference among arguments, we can have that either $KB \models unstable(will)$ or $KB \models risky\_job(will)$, but not both.

A Comparison with Inconsistency-tolerant Semantics

- A variety of inconsistency-tolerant semantics have been developed in the last decade for ontological languages, including lightweight Description Logics (DLs), such as $\mathcal{EL}$ and *DL-Lite* and several fragments of Datalog$^{\pm}$ [7].

- We now analyze entailment in defeasible Datalog$^{\pm}$ ontologies in relation to several inconsistency-tolerant semantics for ontological languages:

  – *AR* (*ABox Repair*) semantics [6];

  – *CAR* (*Closed ABox Repair*) semantics [6];

  – *IAR* (*Intersection ABox Repair*), *k*-support [3], and *ICAR* (*Intersection Closed ABox Repair*) semantics that are sound approximations of *AR* and of *CAR*, respectively; and finally,

  – *k*-defeater semantics [2] that comprises a family of complete approximations of *AR*.

Repairs

- We present the basic concepts needed to understand the different semantics on Datalog$^{\pm}$ ontologies and then show how entailment under such semantics compare to entailment on defeasible Datalog$^{\pm}$.
- We first recall the notion of a *repair*; in relational databases a repair is a model of the set of integrity constraints that is maximally close, *i.e.*, "*as close as possible*" to the original database.
- Different notions of repairs have been developed depending on the meaning of "closeness" used and on the type of constraints.
- For a Datalog$^{\pm}$ ontology $KB = (I, \Sigma_T \cup \Sigma_{NC})$, repairs are maximal subsets of I such that their consequences with respect to Σ_T are non-conflicting relative to Σ_{NC}.

Repairs (Cont.)

- Let the *consistent closure* of an ontology be the set $CCL(KB)$ of formulas in the closure of a consistent subset of the ontology.
- A *Closed ABox repair* of a Datalog$^{\pm}$ ontology KB is a consistent subset I' of $CCL(KB)$ such that there is no other consistent set $I'' \subseteq \mathcal{CLC}(KB)$ that is either $I'' \cap I \supsetneq I' \cap I$ or $I'' \cap I = I' \cap I$ and $I'' \supsetneq I'$.
- Intuitively, a *CAR* repair is a subset of the consistent closure of the knowledge base maximally preserving the database instance [6].

AR and CAR semantics

- **AR Semantics:** The *AR* semantics is the generalization to the notion of *consistent answers* in relational databases [1].

  Intuitively, an atom L is said to be *AR*-consistently entailed from KB, denoted $KB \models_{AR} L$ iff L is classically entailed from every ontology that can be built from every possible repair.

- **CAR Semantics:** An atom L is *CAR*-consistently entailed from KB, denoted by $KB \models_{CAR} L$ iff L is classically entailed from every ontology built from each possible closed ABox repair.

AR and CAR semantics and Defeasible Datalog$^\pm$

- Every atom that is *AR*-consistently (resp., *CAR*-consistently) entailed from a Datalog$^\pm$ ontology $KB = (I, \Sigma_T \cup \Sigma_{NC})$ is also entailed from the defeasible Datalog$^\pm$ ontology $KB' = (\emptyset, I, \Sigma_T, \emptyset, \Sigma_{NC})$ constructed from *KB*.

- This transformation from Datalog$^\pm$ to defeasible Datalog$^\pm$ is without loss of generality; the inconsistency-tolerant semantics that we study here assume that the knowledge contained in I is somehow *challengeable* as it can be in conflict once considered together with the set of constraints.

- Thus, to make a fair comparison between approaches we need to translate the data contained in I to defeasible atoms.

AR and CAR semantics and Defeasible Datalog$^{\pm}$

Theorem

Let $KB = (I, \Sigma_T \cup \Sigma_{NC})$ be a Datalog$^{\pm}$ ontology, $KB' = (\emptyset, I, \Sigma_T, \emptyset, \Sigma_{NC})$ be a defeasible Datalog$^{\pm}$ ontology and $\mathfrak{F} = \langle \mathcal{L}_{\mathcal{R}}, \mathbb{A}_{KB'}, \succ \rangle$. Then, (i) if $KB \models_{AR} L$ then $KB' \models_{\mathfrak{F}} L$, and (ii) if $KB \models_{CAR} L$ then $KB' \models_{\mathfrak{F}} L$.

- The converse does not hold. In our running example, *unstable*(*will*) is not entailed by *AR* or by *CAR*, since every (closed) ABox repair either entails *unstable*(*will*) or *risky_job*(*will*), but not both.
- However, it is entailed from *KB* as $\mathcal{A} \succ \mathcal{B}$. Depending on the preference criterion, it could be the case that *unstable*(*will*) would not be entailed, in which case *risky_job*(*will*) could be, or neither would be.
- These results directly extends to *IAR* and *ICAR*, the sound approximations for *AR* and *CAR* defined in [6], and the family of *k*-support semantics from [3].

k-defeater Semantics and Defeasible Datalog$^{\pm}$

- Given a Datalog$^{\pm}$ ontology $KB = (I, \Sigma_T \cup \Sigma_{NC})$, an atom L is entailed from KB under the k-defeater semantics, $KB \models_{k\text{-}def} L$, for some $k \geq 0$, iff no set of facts with size smaller than k is such that it *contradicts* every minimal set from I that yields L.

- From an argumentation point of view, this semantics looks for counter-arguments for L up to a certain size – the size of an argument being the number of (defeasible) atoms used. If no such argument can be found, L is entailed from KB.

- Conflicting atoms could be entailed from KB for any k (except for that in which converges to AR), therefore it does not enjoy the NCE property.

k-defeater Semantics and Defeasible Datalog$^\pm$

In the following, consider $\succ_{\textit{k-def}}$, a preference criterion such that $\mathcal{A} \succ_{\textit{k-def}} B$ iff the number of facts and defeasible atoms used in $\mathcal{A}$ is less than or equal to k, for any $k \geq 0$.

Theorem

Let $KB = (I, \Sigma_T \cup \Sigma_{NC})$ be a Datalog$^\pm$ ontology, $KB' = (\emptyset, I, \Sigma_T, \emptyset, \Sigma_{NC})$ be a defeasible Datalog$^\pm$ ontology, $\mathfrak{F} = \langle \mathcal{L}_{\mathcal{R}}, \mathbb{A}_{KB'}, \succ \rangle$, and $\mathfrak{F}' = \langle \mathcal{L}_{\mathcal{R}}, \mathbb{A}_{KB'}, \succ_{\textit{k-def}} \rangle$, Then,

(i) *if $KB' \models_{\mathfrak{F}} L$ then $KB \models_{\textit{0-def}} L$, and*

(ii) *for any $0 \leq k < k'$ if $KB' \models_{\mathfrak{F}'} L$ then $KB \models_{\textit{k-def}} L$, where k' is the point where AR and k-defeater semantics coincide.*

k-defeater Semantics and Defeasible Datalog$^{\pm}$

- Statement (i) shows, unsurprisingly, that independently of the preference criterion defined over $\mathbb{A}_{KB'}$, $\models_{\mathfrak{F}}$ is a sound approximation to 0-defeaters. Furthermore, this approximation is not only sound but it also satisfies the NCE property.
- Statement (ii) shows that, using $\succ_{k\text{-}def}$, we have that:

Property

Argumentation-based entailment on defeasible Datalog$^{\pm}$ is a sound approximation of the k-defeater semantics for every k (up to the point where k-defeater coincides with AR).

Property

We have obtained a family of semantics that soundly approximate the k-defeater semantics and ensure that no conflicting atoms can be entailed from a defeasible ontology (as the NCE property holds independently of the preference criterion).

Conclusions

- We have introduced the idea of defeasible reasoning over Datalog$^{\pm}$ ontologies.
  - We introduce the construction of arguments,
  - we provide a way to determine conflicts using the negative constrains available in the system,
  - and complete the argumentative infrastructure by characterizing defeat employing a preference criterion that has remained as an abstract element to be instantiated.
- The dialectical process to decide what arguments are warranted is done applying the classic techniques of argumentation theory.
- Furthermore, we show how such approach ensures the reasonableness of the answers given by it, as no conflicting atoms can be entailed/warranted in it.

Conclusions

- We have also shown that atoms entailed from a Datalog$^{\pm}$ ontology, under well-known inconsistency-tolerant semantics, namely *AR* and *CAR* semantics, and sound approximations of these, are also entailed from the corresponding defeasible Datalog$^{\pm}$ ontology that includes the database instance of the ontology as defeasible atoms.
- Moreover, we have shown that the converse property does not hold in general, and therefore argumentation-based query answering for defeasible Datalog$^{\pm}$ ontologies allows to produce answers that, although are involved in conflicts, and therefore are not consistent answers the ontology contains enough information in order to warrant them.
- Furthermore, we show how to construct a Datalog$^{\pm}$ argumentation framework that yields a semantics that is a sound approximation to the *k*-defeaters semantics from [2], that enjoys the property of never entailing conflicting atoms.

Acknowledgments

This work was partially supported by

- UK Engineering and Physical Sciences Research Council (EPSRC) grant EP/J008346/1 (Probabilistic Ontological Query Answering on the Web - PrOQAW), Department of Computer Science, University of Oxford,
- Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET) in Argentina,
- Secretaría General de Ciencia y Tecnología (SeGCyT) from Universidad Nacional del Sur in Bahia Blanca, Argentina.

References I

M. Arenas, L. E. Bertossi, and J. Chomicki.
Consistent query answers in inconsistent databases.
In *Proc. of PODS*, pages 68–79, 1999.

M. Bienvenu.
On the complexity of consistent query answering in the presence of simple ontologies.
In *Proc. of AAAI*, 2012.

M. Bienvenu and R. Rosati.
Tractable approximations of consistent query answering for robust ontology-based data access.
In *Proc. of IJCAI*, 2013.

References II

A. Calì, G. Gottlob, and T. Lukasiewicz.
A general Datalog-based framework for tractable query answering over ontologies.
In *J. Web Sem.*, volume 14, pages 57–83, 2012.

A. J. García and G. R. Simari.
Defeasible logic programming: An argumentative approach.
TPLP, 4(1-2):95–138, 2004.

D. Lembo, M. Lenzerini, R. Rosati, M. Ruzzi, and D. F. Savo.
Inconsistency-tolerant semantics for description logics.
In *Proc. of RR*, pages 103–117, 2010.

T. Lukasiewicz, M. V. Martinez, and G. I. Simari.
Inconsistency handling in Datalog+/– ontologies.
In *Proc. of ECAI*, pages 558–563, 2012.

References III

Maria Vanina Martinez, Cristhian A. D. Deagustini, Marcelo A. Falappa, and Guillermo R. Simari.
Inconsistency-Tolerant Reasoning in Datalog$^{\pm}$.
In *Advances in Artificial Intelligence-IERAMIA 2014*, pages 15–27, 2014.

Thank you!

Comments? Questions?