On Conceptual Graphs and Explanation of Query Answering Under Inconsistency

Abdallah Arioua, Nouredine Tamani, Madalina Croitoru

On Conceptual Graphs and Explanation of Query Answering Under Inconsistency

Abdallah Arioua, Nouredine Tamani, Madalina Croitoru

University of Montpellier II, France
{arioua,tamani,croitoru}@lirmm.fr

Abstract. Conceptual Graphs are a powerful visual knowledge representation language. In this paper we are interested in the use of Conceptual Graphs in the setting of Ontology Based Data Access, and, more specifically, in reasoning in the presence of inconsistency. We present different explanation heuristics of query answering under inconsistency and show how they can be implemented under the Conceptual Graphs editor COGUI.

1 Introduction

We place ourselves in a Rule-based Data Access (RBDA) setting that investigates how to query multiple data sources defined over the same ontology represented using a rule based language. The RBDA is a specific case of the Ontology Based Data Access (OBDA) setting. In RBDA we assume that the ontology is encoded using rules. The growing number of distinct data sources defined under the same ontology makes OBDA an important and timely problem to address. The input to the problem is a set of facts, an ontology and a conjunctive query. We aim to find if there is an answer to the query in the facts (eventually enriched by the ontology).

More precisely, the RBDA problem stated in reference to the classical *forward chaining* scheme is the following: "Can we find an answer to the query Q in a database F' that is built from F by adding atoms that can be logically deduced from F and the rule based ontology $\mathcal{R}$?"

In certain cases, the integration of factual information from various data sources may lead to inconsistency. A solution is then to construct maximal (with respect to set inclusion) consistent subsets of $\mathcal{F}$ called *repairs* [6, 20]. Once the repairs are computed, there are different ways to combine them in order to obtain an answer for the query.

In this paper we address the RBDA problem from a Conceptual Graphs perspective. Conceptual Graphs are powerful visual formalism for representing a subset of First Order Logic covered by the RBDA setting.

We make explicit these links and focus on the case where we want to perform query answering in presence of inconsistency. We present query answering explanation strategies inspired from the link between the ODBA inconsistent-tolerant semantics and argumentation acceptance semantics[11]. Our work is inspired by the argumentation explanation power [14, 25, 28].

2 Related work

There are two major approaches in order to represent an ontology for the OBDA problem and namely Description Logics (such as $\mathcal{EL}$([2]) and DL-Lite [9] families) and rule based languages (such as the Datalog$^+$ [8] language, a generalization of Datalog that allows for existentially quantified variables in the head of the rules). When using rules for representing the ontology we would denote the OBDA problem under the name of RBDA. Despite Datalog$^+$ undecidability when answering conjunctive queries, there exist decidable fragments of Datalog$^+$ which are studied in the literature [5]. These fragments generalize the above mentioned Description Logics families.

Here we follow the second method: representing the ontology via rules. We give a general rule based setting knowledge representation language equivalent to the Datalog$^+$ language and show how this language is equivalent to Conceptual Graphs with rules and negative constraints.

Within this language we are mainly interested in studying the question of "why an *inconsistent* KB entails a certain query α under an inconsistency-tolerant semantics". Indeed, many works focused on the following questions: "Why a concept C is subsumed (non-subsumed) by D" or "Why the KB is unsatisfiable and incoherent"? The need for explanation-aware methods stems from the desire to seek a comprehensive means that facilitates maintenance and repairing of inconsistent knowledge bases as well as understanding the underlying mechanism for reasoning services. In the field of databases there has been work on explaining answer and non-answer returned by database systems [1, 24, 23, 16, 15] using causality and responsibility or using a cooperative architecture to provide a cooperative answer for query failing.

In the area of DLs, the question was mainly about explaining either reasoning (subsumption and non-subsumption) or unsatisfiability and incoherence. In a seminal paper McGuinness et al. [22, 7] addressed the problem of explaining subsumption and non-subsumption in a coherent and satisfiable DL knowledge base using *formal proofs as explanation* based on a complete and sound deduction system for a fragment of Description Logics, while other proposals [27, 26, 3, 4] have used *Axiom pinpointing* and *Concept pinpointing* as explanation to highlight contradictions within an unsatisfiable and incoherent DL KB.

Another proposal [19, 18] is the so-called *justification-oriented proofs* in which the authors proposed a *proof-like* explanation without the need for deduction rules. The explanation then is presented as an acyclic proof graph that relates axioms and lemmas. Another work [12] in the same context proposes a resolution-based framework in which the explanation is constructed from a refutation graph.

3 Logical Language

We consider a (potentially inconsistent) knowledge base composed of the following:

- A set $\mathcal{F}$ of facts that correspond to existentially closed conjunctions of atoms. The atoms can contain n-ary predicates. The following facts are borrowed from [21]: $F_1 : directs(John, d_1)$, $F_2 : directs(Tom, d_1)$, $F_3 : directs(Tom, d_2)$, $F_4 : supervises(Tom, John)$, $F_5 : works\_in(John, d_1)$, $F_6 : works\_in(Tom, d_1)$.

- A set of negative constraints which represent the negation of a fact. Alternatively negative constraints can be seen as rules with the absurd conclusion. Negative constraints can also be n-ary. For example, $N_1 = \forall x, y, z\,(supervises(x, y) \wedge work\_in(x, z) \wedge directs(y, z)) \rightarrow \bot$ and
  $N_2 = \forall x, y\, supervises(x, y) \wedge manager(y) \rightarrow \bot$ are negative constraints.
- An ontology composed of a set of rules that represent general implicit knowledge and that can introduce new variables in their head (conclusion). Please note that these variables, in turn, can trigger new rule application and cause the undecidability of the language in the general case. Different rule application strategies (chase), including the skolemized chase, are studied in the literature. For example
  $R_1 = \forall x \forall d\, works\_in(x, d) \rightarrow emp(x)$
  $R_2 = \forall x \forall d\, directs(x, d) \rightarrow emp(x)$
  $R_3 = \forall x \forall d\, directs(x, d) \wedge works\_in(x, d) \rightarrow manager(x)$
  $R_4 = \forall x\, emp(x) \rightarrow \exists y(\textit{office}(y) \wedge uses(x, y))$

A rule is *applicable* to set of facts $\mathcal{F}$ if and only if the set entails the hypothesis of the rule. If rule R is applicable to the set F, the application of R on F produces a new set of facts obtained from the initial set with additional information from the rule conclusion. We then say that the new set is an *immediate derivation* of F by R denoted by $R(F)$. For example $R_1(F_5) = works\_in(John, d_1) \wedge emp(John)$.

Let F be a set of facts and let $\mathcal{R}$ be a set of rules. A set F_n is called an $\mathcal{R}$*-derivation* of F if there is a sequence of sets (*derivation sequence*) $(F_0, F_1, \ldots, F_n)$ such that: (i) $F_0 \subseteq F$, (ii) F_0 is $\mathcal{R}$-consistent, (iii) for every $i \in \{1, \ldots, n-1\}$, it holds that F_i is an immediate derivation of F_{i-1}.

Given a set $\{F_0, \ldots, F_k\}$ and a set of rules $\mathcal{R}$, the closure of $\{F_0, \ldots, F_k\}$ with respect to $\mathcal{R}$, denoted $\mathtt{Cl}_{\mathcal{R}}(\{F_0, \ldots, F_k\})$, is defined as the smallest set (with respect to $\subseteq$) which contains $\{F_0, \ldots, F_k\}$, and is closed for $\mathcal{R}$-derivation (that is, for every $\mathcal{R}$-derivation F_n of $\{F_0, \ldots, F_k\}$, we have $F_n \subseteq \mathtt{Cl}_{\mathcal{R}}(\{F_0, \ldots, F_k\})$). Finally, we say that a set $\mathcal{F}$ and a set of rules $\mathcal{R}$ *entail* a fact G (and we write $\mathcal{F}, \mathcal{R} \models G$) iff the closure of the facts by all the rules entails F (i.e. if $\mathtt{Cl}_{\mathcal{R}}(\mathcal{F}) \models G$).

Given a set of facts $\{F_1, \ldots, F_k\}$, and a set of rules $\mathcal{R}$, the set of facts is called $\mathcal{R}$*-inconsistent* if and only if there exists a constraint $N = \neg\mathcal{F}$ such that $\mathtt{Cl}_{\mathcal{R}}(\{F_1, \ldots, F_k\}) \models \mathcal{F}$. A set of facts is said to be $\mathcal{R}$*-consistent* if and only if it is not $\mathcal{R}$-inconsistent.

A knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$, composed of a set of facts (denoted by $\mathcal{F}$), a set of rules (denoted by $\mathcal{R}$) and a set of negative constraints (denoted by $\mathcal{N}$), is said to be *consistent* if and only if $\mathcal{F}$ is $\mathcal{R}$-consistent. A knowledge base is *inconsistent* if and only if it is not consistent.

The above facts $\{F_1, ..., F_6\}$ are $\mathcal{R}$-inconsistent with $\mathcal{R} = \{R_1, ..., R_4\}$ since $\{F_1, F_4, F_6\}$ activate together N_1. Moreover, R_3 can be applied on F_1 and F_5 delivering the new fact $manager(John)$ which put together with F_4 activate N_2.

3.1 Conceptual Graphs Representation

Conceptual Graphs are a visual, logic-based knowledge representation knowledge representation formalism. They encode a part of the ontological knowledge in a structure called *support*. The support consists of a number of taxonomies of the main concepts

(unary predicates) and relations (binary or more predicates) used to describe the world. Note that these taxonomies correspond to certain rules in Datalog. More complex rules (for instance representing transitivity or symmetry of relations) or rules that introduce existential variables in the conclusion are represented using Conceptual Graphs rules. Finally, negative constraints represent rules with the conclusion the absurd operator (or, logically equivalent, negation of facts).

The factual information is described using a bipartite graph in which the two classes of the partition are the concepts, and the relations respectively.

We recall the definition of support and fact following [10]. We consider here a simplified version of a support $S = (T_C, T_R, \mathcal{I})$, where: $(T_C, \leq)$ is a finite partially ordered set of *concept types*; $(T_R, \leq)$ is a partially ordered set of *relation types*, with a specified *arity*; $\mathcal{I}$ is a set of *individual markers*. A (simple) CG is a triple CG$= [S, G, \lambda]$, where:

- S is a support;
- $G = (V_C, V_R, E)$ is an ordered bipartite graph ; $V = V_C \cup V_R$ is the node set of G, V_C is a finite nonempty set of *concept nodes*, V_R is a finite set of *relation nodes*; E is the set of edges $\{v_r, v_c\}$ where the edges incident to each relation node are ordered and this ordering is represented by a positive integer label attached to the edge; if the edge $\{v_r, v_c\}$ is labeled i in this ordering then v_c is the i-neighbor of v_r and is denoted by $N_G^i(v_r)$;
- $\lambda : V \to S$ is a labeling function; if $v \in V_C$ then $\lambda(v) = (type_v, ref_v)$ where $type_v \in T_C$ and $ref_v \in \mathcal{I} \cup \{*\}$; if $r \in V_R$ then $\lambda(r) \in T_R$.

We denote a conceptual graph CG$= [S, G, \lambda]$ by G, keeping support and labeling implicit. The order on $\lambda(v)$ preserves the (pair-wise extended) order on T_C (T_R), considers $\mathcal{I}$ elements mutually incomparable, and $* \geq i$ for each $i \in \mathcal{I}$. Usually, CGs are provided with a logical semantics via the function Φ, which associates to each CG a FOL formula (Sowa (1984)). If S is a support, a constant is associated to each individual marker, a unary predicate to each concept type and a n-ary predicate to each n-ary relation type. We assume that the name for each constant or predicate is the same as the corresponding element of the support. The partial orders specified in S are translated in a set of formulae $\Phi(S)$ by the following rules: if $t_1, t_2 \in T_C$ such that $t_1 \leq t_2$, then $\forall x(t_2(x) \to t_1(x))$ is added to $\Phi(S)$; if $t_1, t_2 \in T_R$, have arity k and $t_1 \leq t_2$, then $\forall x_1 \forall x_2 \ldots \forall x_k(t_2(x_1, x_2, \ldots, x_k) \to t_1(x_1, x_2, \ldots, x_k))$ is added to $\Phi(S)$.

If CG$= [S, G, \lambda]$ is a conceptual graph then a formula Φ(CG) is constructed as follows. To each concept vertex $v \in V_C$ a term a_v and a formula $\phi(v)$ are associated: if $\lambda(v) = (type_v, *)$ then $a_v = x_v$ (a logical variable) and if $\lambda(v) = (type_v, i_v)$, then $a_v = i_v$ (a logical constant); in both cases, $\phi(v) = type_v(a_v)$. To each relation vertex $r \in V_R$, with $\lambda(r) = type_r$ and $deg_G(r) = k$, the formula associated is $\phi(r) = type_r(a_{N_G^1(r)}, \ldots, a_{N_G^k(r)})$.

Φ(CG) is the existential closure of the conjunction of all formulas associated with the vertices of the graph. That is, if $V_C(*) = \{v_{i_1}, \ldots, v_{i_p}\}$ is the set of all concept vertices having generic markers, then Φ(CG)$= \exists v_1 \ldots \exists v_p(\wedge_{v \in V_C \cup V_R} \phi(v))$.

If (G, λ_G) and (H, λ_H) are two CGs (defined on the same support S) then $G \geq H$ (G subsumes H) if there is a *projection* from G to H. A projection is a mapping π

from the vertices set of G to the vertices set of H, which maps concept vertices of G into concept vertices of H, relation vertices of G into relation vertices of H, preserves adjacency (if the concept vertex v in V_C^G is the ith neighbour of relation vertex $r \in V_R^G$ then $\pi(v)$ is the ith neighbour of $\pi(r)$) and furthermore $\lambda_G(x) \geq \lambda_H(\pi(x))$ for each vertex x of G. If $G \geq H$ then $\Phi(S), \Phi(H) \models \Phi(G)$ (*soundness*). *Completeness* (if $\Phi(S), \Phi(H) \models \Phi(G)$ then $G \geq H$) only holds if the graph H is in *normal form*, i.e. if each individual marker appears at most once in concept node labels.

A CG rule $(Hyp, Conc)$ expresses implicit knowledge of the form "if *hypothesis* then *conclusion*", where hypothesis and conclusion are both basic graphs. This knowledge can be made explicit by applying the rule to a specific fact: intuitively, when the hypothesis graph is found in a fact, then the conclusion graph can be added to this fact. There is a one to one correspondence between some concept nodes in the hypothesis with concept nodes in the conclusion. Two nodes in correspondence refer to the same entity. These nodes are said to be *connection nodes*. A rule can be represented by a bicolored graph or by a pair of two CGs (represented, for instance, on the right and respectively left hand side of the screen).

A rule R can be applied to a CG H if there is a homomorphism from its hypothesis to H. Applying R to H according to such a homomorphism π consists of "attaching" to H the conclusion of R by merging each connection node in the conclusion with the image by π of the corresponding connection node in the hypothesis. When a knowledge base contains a set of facts (say $\mathcal{F}$) and a set of rules (say $\mathcal{R}$), the query mechanism has to take implicit knowledge coded in rules into account. The knowledge base answers a query Q if a CG F' can be derived from $\mathcal{F}$ using the rules of $\mathcal{R}$ such that Q maps to F'.

We note that using the $(\mathcal{F}, \mathcal{R}, \mathcal{N})$ Datalog$^+$ notation or the rule based Conceptual Graphs with negative constraints has the same logical expressivity. However, the added value of using Conceptual Graphs comes from the visual depiction of the knowledge. This aspect is shown next, where the previous example knowledge base is depicted using COGUI, a Conceptual Graphs editor developed by the LIRMM, University of Montpellier 2.

3.2 COGUI CG Editor

All figures depict graphs drawn using the conceptual graph editor Cogui [1]. CoGui is a Conceptual Graphs editor. Please note that Cogui is also fully integrated with the conceptual graph engine Cogitant [2] to perform reasoning on the above mentioned graphs.

Let us consider again the knowledge base previously considered:

- $\mathcal{F}$: $F_1 : directs(John, d_1)$, $F_2 : directs(Tom, d_1)$, $F_3 : directs(Tom, d_2)$, $F_4 : supervises(Tom, John)$, $F_5 : works\_in(John, d_1)$, $F_6 : works\_in(Tom, d_1)$.
- $N_1 = \forall x, y, z\, (supervises(x, y) \wedge work\_in(x, z) \wedge directs(y, z)) \rightarrow \bot$ and $N_2 = \forall x, y\, supervises(x, y) \wedge manager(y) \rightarrow \bot$.
- The set of rules: $R_1 = \forall x \forall d\, works\_in(x, d) \rightarrow emp(x)$ $R_2 = \forall x \forall d\, directs(x, d) \rightarrow emp(x)$

[1] http://www.lirmm.fr/cogui/

[2] http://cogitant.sourceforge.net/

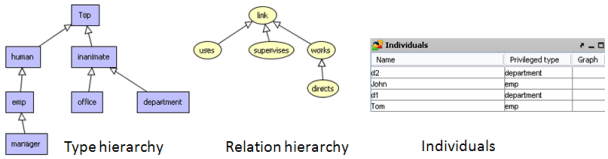

Fig. 1: Visualisation of a support (vocabulary) using CoGui

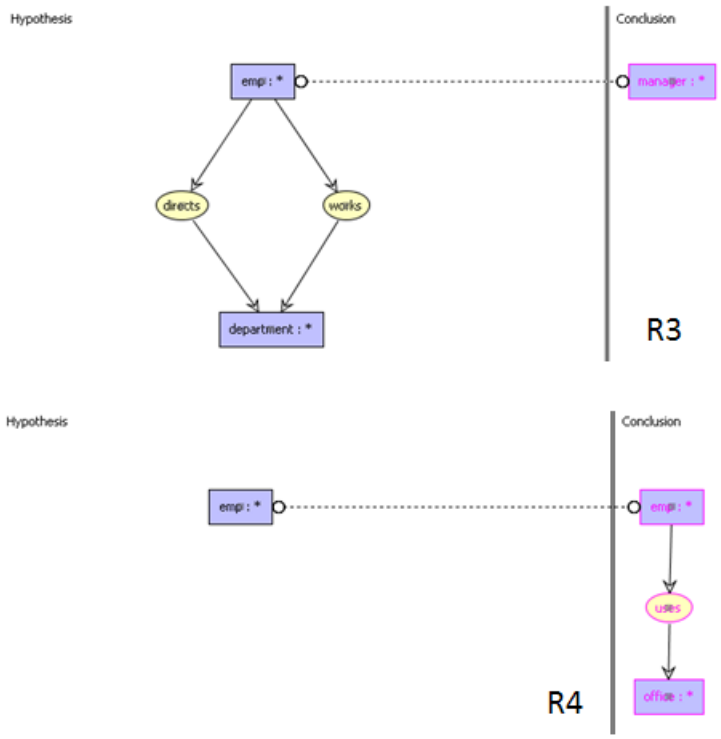

Fig. 2: Visualisation of two rules using CoGui

$$R_3 = \forall x \forall d \, directs(x, d) \wedge works\_in(x, d) \rightarrow manager(x)$$
$$R_4 = \forall x \, emp(x) \rightarrow \exists y (office(y) \wedge uses(x, y))$$

Figure 1 presents the concept type hierarchy, the relation type hierarchy and the list of individuals. Please note that the rule hierarchy encodes the rules R_1 and R_2.

Rules R_3 and R_4 respectively are depicted in Figure 2. The negative constraints N_1 and N_2 are depicted in Figure 3.

Finally, the set of facts is represented in Figure 4.

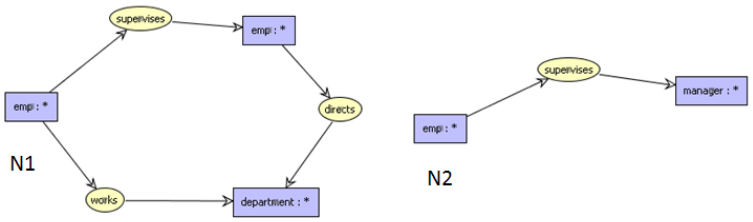

Fig. 3: Visualisation of negative constraints using CoGui

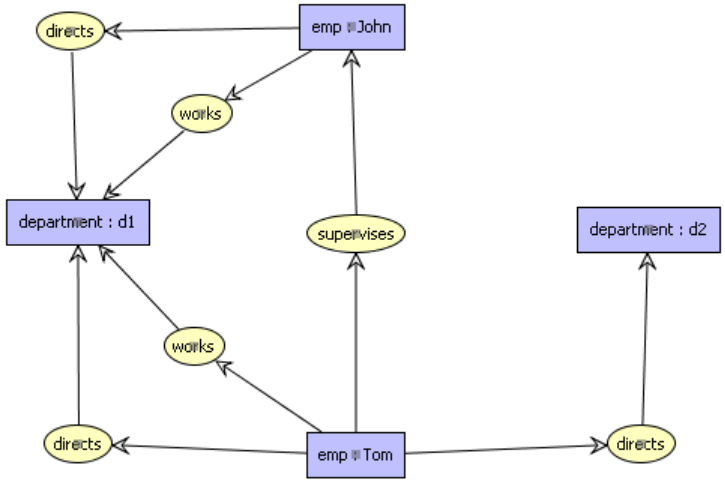

Fig. 4: Visualisation of factual knowledge using CoGui

4 Dealing with inconsistency

We recall the definition of inconsistency. Given a set of facts $\{F_1, \ldots, F_k\}$, and a set of rules $\mathcal{R}$, the set of facts is called $\mathcal{R}$-*inconsistent* if and only if there exists a constraint $N = \neg\mathcal{F}$ such that $\mathtt{Cl}_{\mathcal{R}}(\{F_1, \ldots, F_k\}) \models \mathcal{F}$.

In Figure 5 we can see that there is a negative constraint entailed by the facts enriched by the rules. The image of the negative constraint by homomorphism is represented in red (if color is available) or darker shade of grey (greyscale).

Like in classical logic everything can be entailed from an inconsistent knowledge base. Different semantics have been introduced in order to allow query answering in the presence of inconsistency. Here we only focus on the ICR (**I**ntersection of **C**losed **R**epair) semantics defined as follows:

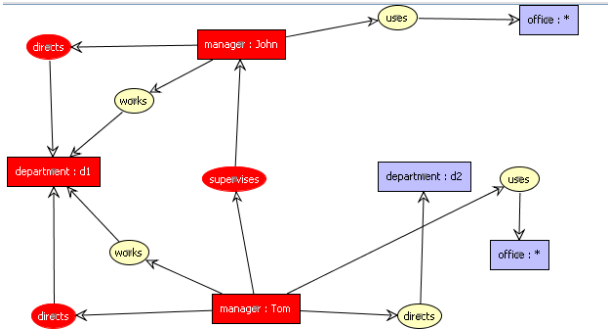

Fig. 5: Visualisation of factual knowledge using CoGui

Definition 1. *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base and let α be a query. Then α is **ICR-entailed** from $\mathcal{K}$, written $\mathcal{K} \models_{ICR} \alpha$ if and only if $\bigcap_{A' \in \mathcal{R}epair(\mathcal{K})} \mathtt{Cl}_{\mathcal{R}}(A') \models \alpha$.*

In the above example, we obtain 6 repairs. The following are one of them (closed under set of rules):

$A_1 = \{directs(John, d_1), directs(Tom, d_1), directs(Tom, d_2),$
$supervises(Tom, John), emp(John), emp(Tom), \exists y_1(office(y_1) \wedge uses(Tom, y_1)),$
$\exists y_2(office(y_2) \wedge uses(John, y_2))\}$

The intersection of the closed repairs is:

$\bigcap_{\mathtt{Cl}_{\mathcal{R}}(A)} = \{directs(Tom, d_1), directs(Tom, d_2), emp(Tom),$
$\exists y\, uses(Tom, y), \exists y office(y)\}$.

Another possibility to deal with an inconsistent knowledge base in the OBDA setting is to define an instantiation [11] of Dung's abstract argumentation theory [13]. An argumentation framework is composed of a set of arguments and a binary relation defined over arguments, the attack.

Definition 2 (Argument). *[11] An argument A in a knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ is a tuple $A = (F_0, \ldots, F_n)$ where:*

- *$(F_0, \ldots, F_{n-1})$ is a derivation sequence w.r.t $\mathcal{K}$.*
- *F_n is an atom, a conjunction of atoms, the existential closure of an atom or the existential closure of a conjunction of atoms such that $F_{n-1} \models F_n$.*

We can extract from each argument its sub-arguments.

Definition 3 (Sub-argument). *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base and $A = (F_0, F_1, \ldots, F_n)$ be an argument. $A' = (F_0, \ldots, F_k)$ with $k \in \{0, ..., n-1\}$ is a sub-argument of A iff (i) $A' = (F_0, \ldots, F_k)$ is an argument and (ii) $F_k \in F_{k+1}$.*

Let $A = (F_0, ..., F_n)$ be an argument, then $\mathtt{Supp}(A) = F_0$ and $\mathtt{Conc}(A) = F_n$. Let $S \subseteq \mathcal{F}$ a set of facts, $Arg(S)$ is defined as the set of all arguments A such that $\mathtt{Supp}(A) \subseteq S$.

An argument corresponds to a rule derivation. Therefore we can use the Cogui editor in order to depict arguments (via the depiction of rule derivations). In Figure 6 an example of a derivation is depicted. The added information by the rule is visible due to the changed color (pink in color, darker shade of grey on grey scale).

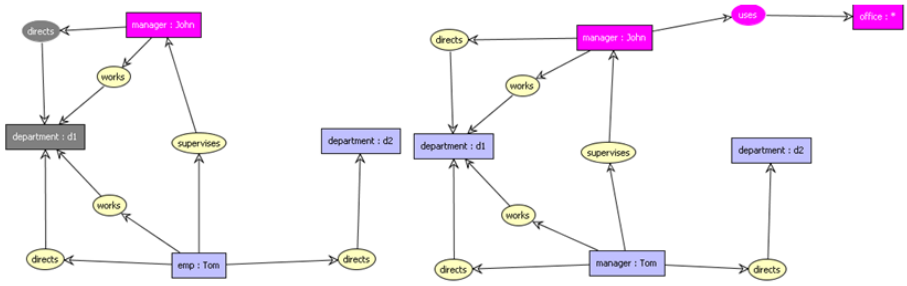

Fig. 6: Visualisation of a rule derivation using CoGui

Definition 4 (Attack). *[11] Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base and let $a, b \in \mathcal{A}$. The argument a attacks b, denoted by $(a, b) \in Att$, iff there exists $\varphi \in \mathtt{Supp}(b)$ such that the set $\{Conc(a), \varphi\}$ is $\mathcal{R}$-inconsistent.*

Definition 5 (Argumentation framework). *[11] Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, the* corresponding argumentation framework *$\mathcal{AF}_K$ is a pair $(\mathcal{A} = \mathtt{Arg}(\mathcal{F}), \mathtt{Att})$ where $\mathcal{A}$ is the set of arguments that can be built from $\mathcal{F}$ and $\mathtt{Att}$ is the attack relation. Let $\mathcal{E} \subseteq \mathcal{A}$ and $a \in \mathcal{A}$.*

- *We say that $\mathcal{E}$ is* conflict free *iff there exists no arguments $a, b \in \mathcal{E}$ such that $(a, b) \in Att$.*
- *$\mathcal{E}$* defends *a iff for every argument $b \in \mathcal{A}$, if we have $(b, a) \in Att$ then there exists $c \in \mathcal{E}$ such that $(c, b) \in Att$.*
- *$\mathcal{E}$ is* admissible *iff it is conflict free and defends all its arguments.*
- *$\mathcal{E}$ is a* preferred extension *iff it is maximal (with respect to set inclusion) admissible set.*
- *$\mathcal{E}$ is a* stable extension *iff it is conflict-free and $\forall a \in \mathcal{A} \backslash \mathcal{E}$, there exists an argument $b \in \mathcal{E}$ such that $(b, a) \in Att$.*
- *$\mathcal{E}$ is a* grounded extension *iff $\mathcal{E}$ is a minimal (for set inclusion) complete extension.*

We denote by $\mathtt{Ext}(\mathcal{AF}_K)$ the set of extensions of $\mathcal{AF}_K$. We use the abbreviations p, s, and g for respectively preferred, stable and grounded semantics. An argument is skeptically accepted if it is in all extensions, credulously accepted if it is in at least one extension and rejected if it is not in any extension.

The following results are then showed by [11]:

Theorem 1. *[11] Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, let $\mathcal{AF}_K$ be the corresponding argumentation framework, α be a query, and $x \in \{s, p\}$ be stable or preferred semantics. Then $\mathcal{K} \models_{ICR} \alpha$ iff α sceptically accepted under semantics x.*

5 Argumentative Explanation

In this section we define two different heuristics for explanation of inconsistency tolerant semantics. Since these heuristics work under inconsistent knowledge bases the Cogui editor is not yet adapted to implement them. We note that explanations correspond to the notion of argument, thus, the Cogui visual power could be easily adapted for our case. Moreover, in section 5.1 we show the equivalence between one type of explanation and a visual rule depiction in Cogui. This could be a starting point for the explanation of queries under inconsistency using Cogui.

When handling inconsistent ontological knowledge bases we are interested in the *explanation* of query answers conforming to a given semantics. More precisely we are interested in explaining why a query α is ICR-entailed by an inconsistent knowledge base $\mathcal{K}$. By explanation we mean a *structure* that has to incorporate minimal set of facts (w.r.t $\subseteq$) and general rules that, if put together, will lead to the entailment of the query α. According to this intuition (which coincides with the definition of [17]) and the link between inconsistent ontological knowledge bases and logic-based argumentation framework, a first candidate of explanation is an argument. However, an argument as defined in definition 2 can be cumbersome and difficult to understand, because the information of how these derivations have been achieved and how they lead to the conclusion are missing. Therefore we propose a refined explanation that incorporates rules as a crucial component.

Definition 6 (Explanation). *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be an inconsistent knowledge base, let α be query and let $\mathcal{K} \models_{ICR} \alpha$. An explanation for α in $\mathcal{K}$ is a 3-tuple $E = (A, G, C)$ composed of three finite sets of formulae such that: (1) $A \subseteq \mathcal{F}$, $G \subseteq \mathcal{R}$, (2) $C \models \alpha$, (3) $\mathtt{Cl}_G(A) \not\models \bot$ (consistency), (4) For every formula β in A, $\mathtt{Cl}_G(A - \beta) \not\models C$ (minimality). Such that $\mathtt{Cl}_G$ represents the closure w.r.t to the set of rules G.*

We denote by $\mathcal{EXP}$ the universe of explanations and by $\mathcal{EXP}_\alpha$ the set of all explanations for α. We denote the sets A, G and C as antecedents, general laws and conclusions respectively. Here the definition specifies three important components for explaining query α. First, the set A of antecedent conditions which is a minimal subset of facts that entails the query α. Second, the set of general laws G (from now on, rules) that produce the query α, the reason for integrating rules is that the user is often interested in knowing how we achieved the query. Finally, the third component is the conclusion C (the answer for the query α). The definition also imposes a central concept, namely *explanation consistency*.

An explanation can be computed directly from $\mathcal{K}$ or from an argumentation framework using the following mapping $\mathbb{A}$.

Definition 7 (Mapping $\mathbb{A}$). *Given an inconsistent knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ and $\mathcal{AF}_K = (\mathcal{A}, \mathtt{Att})$ the corresponding argumentative framework. The mapping $\mathbb{A}$ is a total function defined on $\mathcal{A} \longrightarrow \mathcal{EXP}$ as $E = \mathbb{A}((F_0, ..., F_n))$ with:*

- *The set of antecedent conditions* $A = F_0$,
- *The set of rules* $G \subseteq \mathcal{R}$, *such that* $\forall r \in \mathcal{R}$, $r \in G$ *iff for all* F_i *in* x *the rule* r *is applicable to* F_i.
- *The conclusion* $C = F_n$ *iff* $\mathtt{Cl}_{\mathcal{G}}(A) \models F_n$.

Proposition 1 (Bijection of $\mathbb{A}$). *For any argument $a \in \mathcal{A}$, the mapping $\mathbb{A}$ is a bijection.*

The proposition follows from the definition of the mapping because for every argument we can construct one explanation. Since the mapping is a bijection, we call the argument $x_e = \mathbb{A}^{-1}(e)$ the corresponding argument of an explanation e. We say the argument x_e supports the explanation e. The following proposition states that there is always an explanation for an ICR-entailed query.

Proposition 2 (Existence of explanation). *For every query α such that $\mathcal{K} \models_{ICR} \alpha$, the set $\mathcal{EXP}_\alpha$ is not empty.*

Proof 1 *On the one hand, if $\mathcal{K} \models_{ICR} \alpha$ then the query α is sceptically accepted. That means $\forall \mathcal{E} \in \mathtt{Ext}(\mathcal{AF}_K), \mathcal{E} \models \alpha$. Hence there is an argument $a \in \mathcal{E}$ such that $Cons(a) \models \alpha$. On the other hand, using the mapping $\mathbb{A}$ we have $e = \mathbb{A}(a)$ is an explanation for α, namely $e \in \mathcal{EXP}_\alpha$. Consequently $\mathcal{EXP}_\alpha \neq \emptyset$*

Example 1 (Corresponding Argument). Let us explain $\alpha = \exists x\, emp(x)$. We can build the following argument for α:

$a^+_\alpha = (\{works\_in(Tom, d_1)\}, \{works\_in(Tom, d_1), emp(Tom)\}, emp(Tom))$,
and the delivered explanation is:

$E_\alpha = (\{directs(Tom, d_1)\}, \{\forall x \forall d\, works\_in(x, d) \rightarrow emp(x)\}, emp(Tom))$.

There could be cases where the user wants to know how the set of *rules and facts* interact in order to explain a query α. Put differently, a user-invoked explanation that makes explicit any relation between the facts and the rules which lead to α. Notice that, this type of user-invoked explanation is called *deepened* explanation and it should not confounded with a proof-like explanation, because we are considering an inconsistent and incomplete settings. For this reason the explanation below has not yet been implemented as a stand alone plugin for Cogui (Cogui only deals with querying consistent knowledge).

5.1 Deepened Explanation (d-explanation)

Definition 8 (d-explanation). *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be an inconsistent knowledge base, let α be a query and let $\mathcal{K} \models_{ICR} \alpha$. Then, the finite sequence of tuples $d = \langle t_1, t_2, ..., t_n \rangle$ is a d-explanation for α iff:*

1. *For every tuple $t_i = (a_i, r_i) \in d$ such that $i \in \{1, ..., n\}$, it holds that $a_i \subseteq \mathtt{Cl}_{\mathcal{R}}(\mathcal{F})$ and $r_i \in \mathcal{R}$.*
2. *For every tuple $t_i = (a_i, r_i) \in d$ such that $i \in \{2, ..., n\}$ we have $a_i = a'_i \cup a''_i$ where (i) $r_{i-1}(a_{i-1}) \models a'_i$, (ii) $a''_i \subseteq \mathtt{Cl}_{\mathcal{R}}(\mathcal{F})$ and (iii) r_i is applicable to a_i. Note that if $i = 1$ then $a'_i = \emptyset$.*
3. *The tuple (a_n, r_n) entails α (i.e $r_n(a_n) \models \alpha$).*

4. $\texttt{Cl}_{\mathcal{R}}(\cup_{i=0}^{n} a_i) \nvDash \bot$ *(consistency).*

We denote by $\mathcal{D}$ the universe of d-explanations and by $\mathcal{D}_\alpha$ the set of all d-explanations for a query α.

The intuition about the d-explanation d is as follows: tuples in d represent $\langle fact, applicable\, rule \rangle$, and the sequence of tuples represents the order by which we achieve the answer of the query. Think of it as a chain where each a_i has a link with the previous a_{i-1} through the rule r_{i-1}. This is similar to the notion of derivation depicted in Figure 6.

Example 2 (Deepened explanation). The deepened explanation associated to α is the same as E and doesn't provide more information. Let us consider the explanation of $\alpha_2 = \exists x\, \textit{office}(x)$. A possible argument for α_2 is:

$a^+_{\alpha_2} = (\{works\_in(Tom, d_1)\}, \{works\_in(Tom, d_1), emp(Tom)\},$
$\{works\_in(Tom, d_1), emp(Tom), \exists y(\textit{office}(y) \wedge uses(Tom, y))\},$
$\exists y(\textit{office}(y) \wedge uses(Tom, y))).$

So $E_{\alpha_2} = (\{works\_in(Tom, d_1)\}, \{\forall x \forall d\, directs(x, d) \rightarrow emp(x), \forall x\, emp(x) \rightarrow$
$\exists y(\textit{office}(y) \wedge uses(x, y))\}, \exists y \textit{office}(y)).$

$D_E = (\langle works\_in(Tom, d_1), \forall x \forall d\, directs(x, d) \rightarrow emp(x) \rangle,$
$\langle emp(Tom), \forall x\, emp(x) \rightarrow \exists y(\textit{office}(y) \wedge uses(x, y)) \rangle).$

There is a bijection between an explanation e and a d-explanation d represented here by the following mapping.

Definition 9 (Mapping $\mathbb{D}$). *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be an inconsistent knowledge base, α be a query, $e = (A, G, C) \in \mathcal{EXP}$ be an explanation for α and $d = \langle t_1, t_2, ..., t_n \rangle \in \mathcal{D}$ be a d-explanation for α. The mapping $\mathbb{D}$ is total function $\mathbb{D} : \mathcal{EXP} \longrightarrow \mathcal{D}$, $e \longrightarrow d$ defined as follows:*

1. *For every tuple $t_i = (a_i, r_i)$ such that $i \in \{1, ..., n\}$, it holds that $r_i \in G$.*
2. *For every tuple $t_i = (a_i, r_i)$ in D such that $i \in \{2, ..., n\}$ we have $a_i = a'_i \cup a''_i$ where $r_{i-1}(a_{i-1}) \vDash a'_i$, $a''_i \subseteq A$ and r_i is applicable to a_i. Note that if $i = 1$ then $a_i = A$ and e_i is applicable to a_i.*
3. *The tuple (a_n, r_n) entails α (i.e $r_n(a_n) \vDash \alpha$) and $C \vDash \alpha$.*

Since the mapping is a bijection the existence of the inverse function is guaranteed. Thereby we consider the mapping $\mathbb{D}(e)$ as deepening the explanation e and the inverse mapping $\mathbb{D}^{-1}(d)$ as simplifying the d-explanation d. The advantage of such a mapping is that it gives the users the freedom to shift from an explanation to another which complies better with their level of understanding and their experiences. Also it guarantees that every explanation can be deepened. As done before, we also define the corresponding argument x_d for a d-explanation d, as the corresponding argument x_e for an explanation $e = \mathbb{D}^{-1}(d)$. This can be achieved by the following composition of function: $x_d = (\mathbb{D} \circ \mathbb{A})^{-1}(d)$.

6 Conclusion

In this paper we have presented an argumentative approach for explaining user query answers in a particular setting, Namely, an inconsistent ontological knowledge base

where inconsistency is handled by inconsistency-tolerant semantics (ICR) and it is issued from the set of facts. In this paper we have exploited the relation between ontological knowledge base and logical argumentation framework to establish different levels of explanation ranging from an explanation based on the notion of argument to a user-invoked explanation called deepened explanation. We have also shown the relation between every type of explanation using a one-to-one correspondence which gives the user the possibility to deepen (or simplify) the explanation in hand. Future works aims at studying the proposed explanation in the context of other inconsistency-tolerant semantics. We are currently working on a Cogui based plug-in that only deals with reasoning under inconsistency and the above mentioned semantics.

7 Acknowledgments

A. Arioua and M. Croitoru have been supported by the French National Agency of Research within Dur-Dur project.

References

1. T. Arora, R. Ramakrishnan, W. Roth, P. Seshadri, and D. Srivastava. Explaining program execution in deductive systems. In S. Ceri, K. Tanaka, and S. Tsur, editors, *Deductive and Object-Oriented Databases*, volume 760 of *Lecture Notes in Computer Science*, pages 101–119. Springer Berlin Heidelberg, 1993.
2. F. Baader, S. Brandt, and C. Lutz. Pushing the el envelope. In *Proc. of IJCAI 2005*, 2005.
3. F. Baader, R. Peñaloza, and B. Suntisrivaraporn. Pinpointing in the description logic el. In J. Hertzberg, M. Beetz, and R. Englert, editors, *KI 2007: Advances in Artificial Intelligence*, volume 4667 of *Lecture Notes in Computer Science*, pages 52–67. Springer Berlin Heidelberg, 2007.
4. F. Baader and B. Suntisrivaraporn. Debugging snomed ct using axiom pinpointing in the description logic el+. In *KR-MED*, volume 410, 2008.
5. J.-F. Baget, M.-L. Mugnier, S. Rudolph, and M. Thomazo. Walking the complexity lines for generalized guarded existential rules. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence, (IJCAI'11)*, pages 712–717, 2011.
6. M. Bienvenu. On the complexity of consistent query answering in the presence of simple ontologies. In *Proc of AAAI*, 2012.
7. A. Borgida, E. Franconi, I. Horrocks, D. L. McGuinness, and P. F. Patel-Schneider. Explaining ALC subsumption. In P. Lambrix, A. Borgida, M. Lenzerini, R. Möller, and P. F. Patel-Schneider, editors, *Proceedings of the 1999 International Workshop on Description Logics 1999*, volume 22, Linköping, Sweden, July 1999.
8. A. Calì, G. Gottlob, and T. Lukasiewicz. A general datalog-based framework for tractable query answering over ontologies. In *Proceedings of the Twenty-Eigth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 77–86. ACM, 2009.
9. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The dl-lite family. *J. Autom. Reasoning*, 39(3):385–429, 2007.
10. M. Chein and M.-L. Mugnier. *Graph-based Knowledge Representation and Reasoning—Computational Foundations of Conceptual Graphs*. Advanced Information and Knowledge Processing. Springer, 2009.

11. M. Croitoru and S. Vesic. What can argumentation do for inconsistent ontology query answering? In W. Liu, V. Subrahmanian, and J. Wijsen, editors, *Scalable Uncertainty Management*, volume 8078 of *Lecture Notes in Computer Science*, pages 15–29. Springer Berlin Heidelberg, 2013.
12. X. Deng, V. Haarslev, and N. Shiri. A framework for explaining reasoning in description logics. In *IN: PROCEEDINGS OF THE AAAI FALL SYMPOSIUM ON EXPLANATION-AWARE COMPUTING*, pages 189–204. AAAI Press, 2005.
13. P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-persons games. *Artificial Intelligence*, 77(2):321–357, 1995.
14. A. Garcıa, C. I. Chesnevar, N. D. Rotstein, and G. R. Simari. An abstract presentation of dialectical explanations in defeasible argumentation. *ArgNMR07*, pages 17–32, 2007.
15. P. Godfrey. Minimization in cooperative response to failing database queries. *International Journal of Cooperative Information Systems*, 06(02):95–149, 1997.
16. P. Godfrey, J. Minker, and L. Novik. An architecture for a cooperative database system. In W. Litwin and T. Risch, editors, *Applications of Databases*, volume 819 of *Lecture Notes in Computer Science*, pages 3–24. Springer Berlin Heidelberg, 1994.
17. C. G. Hempel and P. Oppenheim. Studies in the logic of explanation. *Philosophy of Science*, 15(2):pp. 135–175, 1948.
18. M. Horridge, B. Parsia, and U. Sattler. Explaining inconsistencies in owl ontologies. In L. Godo and A. Pugliese, editors, *Scalable Uncertainty Management*, volume 5785 of *Lecture Notes in Computer Science*, pages 124–137. Springer Berlin Heidelberg, 2009.
19. M. Horridge, B. Parsia, and U. Sattler. Justification oriented proofs in owl. In *Proceedings of the 9th International Semantic Web Conference on The Semantic Web - Volume Part I*, ISWC'10, pages 354–369, Berlin, Heidelberg, 2010. Springer-Verlag.
20. 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.
21. T. Lukasiewicz, M. V. Martinez, G. I. Simari, et al. Inconsistency handling in datalog+/- ontologies. In *ECAI*, pages 558–563, 2012.
22. D. L. McGuinness and A. T. Borgida. Explaining subsumption in description logics. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'95, pages 816–821, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.
23. A. Meliou, W. Gatterbauer, J. Y. Halpern, C. Koch, K. F. Moore, and D. Suciu. Causality in databases. *IEEE Data Eng. Bull.*, 33(3):59–67, 2010.
24. A. Meliou, W. Gatterbauer, K. F. Moore, and D. Suciu. Why so ? or why no ? functional causality for explaining query answers. In *Proceedings of the Fourth International VLDB workshop on Management of Uncertain Data (MUD 2010) in conjunction with VLDB 2010, Singapore, September 13, 2010*, 2010.
25. S. Modgil and M. Caminada. Proof theories and algorithms for abstract argumentation frameworks. In *Argumentation in artificial intelligence*, pages 105–129. Springer, 2009.
26. S. Schlobach. Debugging and semantic clarification by pinpointing. In *Proceedings of the Second European Conference on The Semantic Web: Research and Applications*, ESWC'05, pages 226–240, Berlin, Heidelberg, 2005. Springer-Verlag.
27. S. Schlobach and R. Cornet. Non-standard reasoning services for the debugging of description logic terminologies. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, pages 355–360, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc.
28. D. Seselja and C. Strasser. Abstract argumentation and explanation applied to scientific debates. *Synthese*, 190(12):2195–2217, 2013.