A Rational Entailment for Expressive Description Logics via Description Logic Programs

Giovanni Casini[1,2(✉)] and Umberto Straccia[1]

[1] ISTI - CNR, Pisa, Italy
{giovanni.casini,umberto.straccia}@isti.cnr.it
[2] CAIR - University of Cape Town, Cape Town, South Africa

Abstract. Lehmann and Magidor's *rational closure* is acknowledged as a landmark in the field of non-monotonic logics and it has also been re-formulated in the context of *Description Logics* (DLs). We show here how to model a rational form of entailment for expressive DLs, such as $\mathcal{SROIQ}$, providing a novel reasoning procedure that compiles a non-monotone DL knowledge base into a *description logic program* (dl-program).

1 Introduction

One of the main non-monotonic formalism, namely Lehmann and Magidor's *rational closure* [23], is acknowledged as a landmark for non-monotonic reasoning due to its logical properties. Rational closure, that falls under the more general class of the rational entailment relations [23], has been proposed in the context of *Description Logics* (DLs) [1], starting from basic DLs, such as $\mathcal{ALC}$ [3,4,8,9,11,17,18], and re-formulated for low-complexity DLs, as $\mathcal{EL}_\bot$ [12,15], as for expressive ones, up to $\mathcal{SROIQ}$ [2].

Here we show an implementation of a rational entailment relation for an expressive DL such as $\mathcal{SROIQ}$ [19]. The main contribution of this paper is that we re-formulate the decision procedure for rational closure by compiling a non-monotone DL knowledge base into a *description logic program* (dl-program) [13]. Dl-programs have been proposed to combine DLs with *Answer Set Programming* [14], an established approach to implement non-monotonic reasoning for rule-based languages. In this way our approach can be easily implemented on top of existing reasoners supporting dl-programs, such as DLV[1].

We proceed as follows. In Sect. 2 we briefly present the logical systems we will refer to in the definition of our method, which is worked out in Sect. 3. Eventually, in Sect. 4 we briefly recall related work and then we conclude.

2 Preliminaries

For the sake of completeness and to ease the reading, we introduce here a minimum of basic notions.

[1] http://www.dlvsystem.com.

E. Jembere et al. (Eds.): SACAIR 2021, CCIS 1551, pp. 177–191, 2022.
https://doi.org/10.1007/978-3-030-95070-5_12

2.1 Description Logic Programs

Normal Logic Programs. Assume a first-order vocabulary $\Phi = \langle \mathtt{P}, \mathtt{C} \rangle$, with $\mathtt{C}$ a set of constants $\{a, b, \dots\}$ and $\mathtt{P}$ a set of predicates $\{p, q, \dots\}$, and let $\mathtt{X}$ be a set of variables $\{x, y, \dots\}$, with $\mathtt{P}, \mathtt{C}, \mathtt{X}$ mutually disjoint. A *term* t is either a variable from $\mathtt{X}$ or a constant from $\mathtt{C}$, and an *atom* is an expression $p(t_1, \dots, t_n)$, where p is a n-ary predicate in $\mathtt{P}$ and each t_i is a term. A *literal* l is an atom or its negation (via connective $\neg$), while a negation-as-failure literal (*NAF-literal*) is of the form $not\ l$, where l is a literal. A *rule* r is an expression of the form ($m \geq k \geq 0$)

$$a \leftarrow b_1, \dots, b_k, not\ b_{k+1}, \dots, not\ b_m\ , \tag{1}$$

where $a, b_1, \dots, b_m$ are literals. Intuitively, a rule has to be read as "if we know that $b_1, \dots, b_k$ are true, but we are not aware that $b_{k+1}, \dots, b_m$ are true, then we can conclude a". We indicate by $H(r)$ (head of r) the literal a, by $B^+(r)$ (positive body of r) the set $\{b_1, \dots, b_k\}$ and by $B^-(r)$ (negative body of r) the set $\{b_{k+1}, \dots, b_m\}$. A *normal program* P is a finite set of rules, while a *positive program* P is a finite set of rules in which $B^-(r) = \emptyset$ for every rule r.

As usual, atoms, literals, rules and programs are considered *ground* if they do not contain any variable. The *Herbrand Universe* of a program P (HU_P) is the set of all the constants that appear in P, while the *Herbrand Base* of P (HB_P) is the set of all the literals that can be constructed from the predicates in P and the constants in HU_P. A ground instance of a rule r is obtained substituting every variable occurring in r with a constant symbol in HU_P, and, given a program P, $ground(P)$ is the set of all ground instances of rules in P.

From the semantics point of view, an interpretation I of a program P is a *consistent* subset of HB_P, *i.e.* $I \subseteq HB_P$ and there is no atom a such that both a and $\neg a$ are in I. The truth value of a literal l is true, false, or unknown in I iff, respectively, $l \in I$, $\neg l \in I$, or $\{l, \neg l\} \cap I = \emptyset$, where $\neg\neg a$ is a. The satisfiability of a program P is reduced to the satisfiability of its rules expressed in ground form: that is, I is a *model* of a program P iff it is a model of $ground(P)$, *i.e.* if $B^+(r) \subseteq I$ and $B^-(r) \cap I = \emptyset$, then $H(r) \in I$ for every rule in $ground(P)$.

In case of a positive program P, the *answer set* of P is the least model of P with respect to set inclusion: the fact that P is positive guarantees the uniqueness of its answer set [13]. If P is not positive, the notion of *answer set* is defined via the so-called *Gelfond-Lifschitz transformation* (see, *e.g.* [14]). Specifically, consider a program P and an interpretation $I \subseteq HB_P$. The *Gelfond-Lifschitz transformation* of P relative to I gives back a positive program P^I, and it is obtained from $ground(P)$ with the following procedure:

- delete from $ground(P)$ every rule r s.t. $B^-(r) \cap I \neq \emptyset$; and
- from the remaining rules delete the negative part of the body.

In this way, we end up with a positive program P^I, and I is an *answer set* for P iff I is the answer set for the positive ground program P^I. We indicate with $ans(P)$ the set of the answer sets of a program P. Eventually, we define a *cautious* (resp., *brave*) consequence relation $\models_c$ ($\models_b$) as follows: $P \models_c l$ ($P \models_b l$) iff the literal l is true in any (some) answer set of P.

Example 1. Let P be a program composed of the following rules:

$$\begin{aligned} feline(a) &\leftarrow \\ feline(b) &\leftarrow \\ big(b) &\leftarrow \\ docile(x) &\leftarrow feline(x), not\ big(x)\ . \end{aligned}$$

Consider the interpretation $I = \{feline(a), feline(b), big(b), docile(a)\}$. Then P^I is defined as follows:

$$\begin{aligned} feline(a) &\leftarrow \\ feline(b) &\leftarrow \\ big(b) &\leftarrow \\ docile(a) &\leftarrow feline(a)\ . \end{aligned}$$

It is straightforwardly verified that I is the least model of P^I and, thus, I is an answer set of P (actually, it is the only one).

Description Logics. We shall refer here to an expressive DLs, namely $\mathcal{SROIQ}$ (for more details about it, we refer the reader to [19]). The $\mathcal{SROIQ}$ signature is composed of a set of *concept names* $\mathcal{A}t = \{A, B, \ldots\}$, a set of *role names* $\mathcal{S} = \{R, S, \ldots\}$, and a set $\mathcal{O}$ of *individuals* $\{a, b, \ldots\}$. The set of roles is $\mathcal{R} = \mathcal{S} \cup \{R^- \mid R \in \mathcal{S}\} \cup \{U\}$, where R^- is the inverse of a role R (R^{--} is R) and U is the universal role. We can also compose the roles in $\mathcal{R}$ into finite chains such as $R_1 \circ \ldots \circ R_n$. The set $\mathcal{C}$ of $\mathcal{SROIQ}$ concepts is defined inductively as:

(i) $\mathcal{A}t \subseteq \mathcal{C}$;
(ii) $\top, \bot \in \mathcal{C}$;
(iii) if $\{a_1, \ldots, a_n\} \subseteq \mathcal{O}$, then $\{a_1, \ldots, a_n\} \in \mathcal{C}$;
(iv) if $C, D \in \mathcal{C}$, then $C \sqcap D, C \sqcup D, \neg C \in \mathcal{C}$;
(v) if $C \in \mathcal{C}, R \in \mathcal{R}$, then $\exists R.C, \forall R.C, \geq_n R.C, \leq_n R.C, \exists R.\texttt{Self} \in \mathcal{C}$.

Condition (iii) indicates that the enumerated sets of individuals (*nominals*) can be used also in the TBox as concepts. An interpretation is a pair $\langle \Delta^\mathcal{I}, \cdot^\mathcal{I} \rangle$, where $\Delta^\mathcal{I}$ is a nonempty set, called *domain*, and the *interpretation function* $\cdot^\mathcal{I}$ assigns to every individual a member of the domain $\Delta^\mathcal{I}$, to every concept name a subset of $\Delta^\mathcal{I}$, and to every role name a subset of $\Delta^\mathcal{I} \times \Delta^\mathcal{I}$. The function $\cdot^\mathcal{I}$ is extended to all the concepts and roles in the following way:

- $\{o_1, \ldots, o_n\}^\mathcal{I} = \{o_1^\mathcal{I}, \ldots, o_n^\mathcal{I}\}$;
- $(C \sqcap D)^\mathcal{I} = C^\mathcal{I} \cap D^\mathcal{I}$;
- $(C \sqcup D)^\mathcal{I} = C^\mathcal{I} \cup D^\mathcal{I}$;
- $(\neg C)^\mathcal{I} = \Delta^\mathcal{I} / C^\mathcal{I}$;
- $(\exists R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \exists y.(x, y) \in R^\mathcal{I} \wedge y \in C^\mathcal{I}\}$;
- $(\forall R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \forall y.(x, y) \in R^\mathcal{I} \rightarrow y \in C^\mathcal{I}\}$;
- $(\geq_n R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \#\{y \mid (x, y) \in R^\mathcal{I} \wedge y \in C^\mathcal{I}\} \geq n\}$;

- $(\leq_n R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#\{y \mid (x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \leq n\}$;
- $(\exists R.\texttt{Self})^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid (x,x) \in R^{\mathcal{I}}\}$;
- $(R^-)^{\mathcal{I}} = \{(a,b) \mid (b,a) \in R\}$;
- $(U)^{\mathcal{I}} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$;
- $(R_1 \circ \ldots \circ R_n)^{\mathcal{I}} = \{(a,b) \mid \exists x_1, \ldots, x_{n-1}.(a,x_1) \in R_1^{\mathcal{I}}, (x_1,x_2) \in R_2^{\mathcal{I}}, \ldots, (x_{n-1},b) \in R_n^{\mathcal{I}}\}$.

where $\#S$ is the cardinality of set $S \subseteq \Delta^{\mathcal{I}}$. A *DL knowledge base* L is a triple $\langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, where $\mathcal{A}$ is an *ABox*, containing information about the individuals, $\mathcal{T}$ is a *TBox*, containing information about the relations between the concepts, and $\mathcal{R}$ is an *RBox*, containing information about the roles. The form of the allowed axioms are described in Table 1, with their respective semantics ($n \geq 1$).

Table 1. Axioms of *ABox*,*TBox* and *RBox*.

	Axiom name	Syntax	Semantics
ABox	Concept membership axiom	$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
	Role membership axiom	$R(a,b)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
TBox	Concept inclusion axiom	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
RBox	Role inclusion axiom	$R_1 \circ \ldots \circ R_n \sqsubseteq S$	$(R_1 \circ \ldots \circ R_n)^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
	Transitivity	$Trans(R)$	$R^{\mathcal{I}}$ is transitive
	Functionality	$Fun(R)$	$R^{\mathcal{I}}$ is a function
	Reflexivity	$Ref(R)$	$R^{\mathcal{I}}$ is reflexive
	Irreflexivity	$Irr(R)$	$R^{\mathcal{I}}$ is irreflexive
	Symmetry	$Sym(R)$	$R^{\mathcal{I}}$ is symmetric
	Asymmetry	$Asy(R)$	$R^{\mathcal{I}}$ is asymmetric
	Disjointness	$Dis(R,S)$	$R^{\mathcal{I}}$ and $S^{\mathcal{I}}$ are disjoint

A RBox has further to comply with an additional syntactical restriction: that is, a RBox has to be *regular*, which essentially prevents a RBox from containing cyclic dependencies among roles that are known to lead to undecidability [20]. For ease of presentation we do not include the definition here and refer the reader to [19, Definition 2] instead. We use $C = D$ as a shorthand of the concept inclusion axiom $\top \sqsubseteq (\neg C \sqcup D) \sqcap (\neg D \sqcup C)$. With $\models$ we denote the classical, monotonic, consequence/entailment relation, which is defined as usual.

Note also that every ABox axiom can be reformulated as an equivalent TBox axiom. In particular, $C(a)$ can be reformulated as $\{a\} \sqsubseteq C$, while $R(a,b)$ is equivalent to $\{a\} \sqsubseteq \exists R.\{b\}$. Consequently, in what follows we will not consider ABoxes.

Description Logic Programs. A *description logic program* (*dl-program*) is composed of a pair $\mathcal{K} = \langle L, P \rangle$, where L is a DL knowledge base and P is a set of *dl-rules* [13], which we are going to specify next. The DL knowledge base L is defined over a vocabulary composed of a set of concept names $\mathcal{A}t$, a set of role names $\mathcal{S}$, and a set $\mathcal{O}$

of individuals, while P is defined over a vocabulary $\Phi = \langle \mathtt{P}, \mathtt{C} \rangle$, with $\mathtt{C}$ a set of constants and $\mathtt{P}$ a set of predicates, and with $\mathtt{X}$ a set of variables. We assume that the predicative part of the two formalisms are independent, that is $\mathcal{A}t \cup \mathcal{S}$ is disjoint from $\mathtt{P}$, while the same domain of individuals is shared, that is $HU_P \subseteq \mathtt{C} \subseteq \mathcal{O}$.

Dl-programs use the notions of *dl-query* and *dl-atom* to be used in rule bodies to express queries to the DL knowledge base L. That is, a *dl-query* $Q(\mathbf{t})$ can have various forms, but to what concerns us, it is sufficient to consider the following ones:

- a concept membership axiom $C(t)$ (so, $\mathbf{t} = t$);
- a role membership axiom $R(t_1, t_2)$ (so, $\mathbf{t} = \langle t_1, t_2 \rangle$).

On the other hand, a *dl-atom* is an expression of the form[2]

$$DL[S_1 \uplus p_1, \ldots, S_m \uplus p_m; Q](\mathbf{t})$$

with $m \geq 0$, where each S_i is either a concept or a role ($S_i \in \mathcal{C} \cup \mathcal{R}$), and each p_i is a predicate symbol from $\mathtt{P}$, unary if S_i is a concept, binary otherwise, and $Q(\mathbf{t})$ is a dl-query. The operator $\uplus$ is functional to the updating of the DL knowledge base L with factual information obtained from the activation of the rules in the program. That is, each $S_i \uplus p_i$ indicates that the extension of S_i is increased by the extension of p_i.

Now, a *dl-rule* r is of the form (1), where any literal $b_1, \ldots, b_m \in B(r)$ may be a dl-atom and a *dl-program* is a pair $\mathcal{K} = \langle L, P \rangle$, where L is a DL knowledge base and P is a set of dl-rules.

From a semantics points of view, for an interpretations $I \subseteq HB_P$, we say that I is a *model* of a ground literal or dl-atom l under L ($I \models_L l$) iff

- if $l \in HB_P$, then $I \models_L l$ iff $l \in I$;
- if l is a ground dl-atom $DL[\lambda, Q](\mathbf{c})$, where $\lambda = S_1 \uplus p_1, \ldots, S_m \uplus p_m$, then $I \models_L l$ iff $L(I; \lambda) \models Q(\mathbf{c})$, where $L(I; \lambda) = L \cup \bigcup_{i=1}^{m} A_i(I)$, with, for $1 \leq i \leq m$, $A_i(I) = \{S_i(\mathbf{e}) \mid p_i(\mathbf{e}) \in I\}$,

As usual, an interpretation I is a *model* of a ground dl-rule r iff $I \models_L l$ for all $l \in B^+(r)$ and $I \not\models_L l$ for all $l \in B^-(r)$ implies $I \models_L H(r)$. I is a *model* of a dl-program $\mathcal{K} = \langle L, P \rangle$ (written $I \models \mathcal{K}$) iff $I \models_L r$ for all $r \in ground(P)$. We say that $\mathcal{K}$ is *satisfiable* if it has a model.

Let $KB = \langle L, P \rangle$ be a dl-program. The *strong dl-transform* of P w.r.t. L and I (denoted sP_L^I) is the set of all dl-rules obtained from $ground(P)$ by deleting

- every dl-rule r s.t. $I \models_L l$ for some $l \in B^-(r)$;
- from the remaining dl-rule r all literals in $B^-(r)$.

Note that (i) $\langle L, sP_L^I \rangle$ has only monotonic dl-atoms and no NAF-literals anymore; and (ii) a *positive* dl-program, if satisfiable, has a least model [13]. Now, a *strong answer set* of $\mathcal{K} = \langle L, P \rangle$ is an interpretation $I \subseteq HB_P$ s.t. I is the least model of $\langle L, sP_L^I \rangle$. We denote with $ans_s(\mathcal{K})$ the set of the strong answer sets of $\mathcal{K}$. l is a *cautious* (*brave*)

[2] The definition given here is again simpler than the original one, as we consider only the form strictly required for our proposal.

consequence of $\mathcal{K}$, indicated as $\mathcal{K} \models_{s,c} l$ ($\mathcal{K} \models_{s,b} l$) iff l is true in every (some) strong answer of $\mathcal{K}$.

Note that given a dl-program $\mathcal{K} = \langle L, P \rangle$ and an answer set I of $\mathcal{K}$, I is a minimal model of $\mathcal{K}$ [13].

Example 2. Consider a dl-program $\mathcal{K} = \langle L, P \rangle$. Let $L = \langle \mathcal{T} \rangle$, with

$$\mathcal{T} = \{\{a\} \sqsubseteq Cat, \{b\} \sqsubseteq Feline, \{b\} \sqsubseteq Big\} ,$$

and consider a dl-program P composed of the following rules:

$$feline(x) \leftarrow DL[Cat](x)$$
$$docile(x) \leftarrow DL[Feline \uplus feline; Feline](x), not\ DL[Big](x) .$$

It can easily be shown that $\mathcal{K}$ has a unique answer set

$$I = \{feline(a), docile(a)\} .$$

In fact, I is the least model of the following set sP_L^I of dl-rules:

$$feline(b) \leftarrow DL[Cat](b)$$
$$feline(a) \leftarrow DL[Cat](a)$$
$$docile(a) \leftarrow DL[Feline \uplus feline; Feline](a) .$$

2.2 Rational Closure for $\mathcal{ALC}$

For convenience, we recap here some salient notions related to *rational closure* (RC) for DLs, specifically for the DL $\mathcal{ALC}$ (see, *e.g.* [8]).

Remark 1. We remind that $\mathcal{ALC}$ concepts are inductively defined as (i) $\mathcal{A}t \subseteq \mathcal{C}$; (ii) $\top, \bot \in \mathcal{C}$; (iii) if $C, D \in \mathcal{C}$, then $C \sqcap D, C \sqcup D, \neg C \in \mathcal{C}$; (iv)] if $C \in \mathcal{C}, R \in \mathcal{R}$, then $\exists R.C, \forall R.C \in \mathcal{C}$.

Now, a *defeasible concept inclusion* axiom is of the form $C \mathrel{\tilde{\sqsubseteq}} D$, where, without loss of generality, C and D are assumed to be atomic concepts or their negation. The expression $C \mathrel{\tilde{\sqsubseteq}} D$ has to be read as 'if an individual falls under the concept C, typically it falls also under the concept D'. A *defeasible* DL knowledge base is a pair $L = \langle \mathcal{T}, \mathcal{D} \rangle$, where $\mathcal{T}$ (the *TBox*) is a finite set of concept inclusion axioms of the form $C \sqsubseteq D$, where C, D are $\mathcal{ALC}$ concepts, and $\mathcal{D}$ (the *DBox*) is a finite set of defeasible concept inclusion of the form $C \mathrel{\tilde{\sqsubseteq}} D$.

We next briefly describe the decision procedure for RC for $\mathcal{ALC}$, referring in particular to the one presented in [3], that in turn has been obtained by refining the one presented in [8]. Consider $L = \langle \mathcal{T}, \mathcal{D} \rangle$. The first step of the procedure is to assign a rank to each defeasible axiom in $\mathcal{D}$. The rank of the defeasible axioms indicates, in case of conflictual information, which axiom is associated to more specific premises, and has the priority over the axioms associated to more general premises. Central to this step is the exceptionality procedure `Exceptional`$(\cdot)$ (see below). The procedure makes use of the notion of *materialisation*, to reduce concept exceptionality checking to entailment checking, were the *materialisation* of $\mathcal{D}$ is defined as $\overline{\mathcal{D}} := \{\neg C \sqcup D \mid C \mathrel{\tilde{\sqsubseteq}} D \in \mathcal{D}\}$.

Procedure Exceptional(L)

Input: A DL knowledge base $L = \langle \mathcal{T}, \mathcal{D} \rangle$
Output: $\mathcal{E} \subseteq \mathcal{D}$

$\mathcal{E} := \emptyset$;
foreach $C \sqsubset\!\!\!\approx D \in \mathcal{D}$ **do**
  if $\mathcal{T} \models \bigsqcap \overline{\mathcal{D}} \sqsubseteq \neg C$ **then**
    $\mathcal{E} := \mathcal{E} \cup \{C \sqsubset\!\!\!\approx D\}$
return $\mathcal{E}$

Procedure ComputeRanking(L)

Input: A DL knowledge base $L = \langle \mathcal{T}, \mathcal{D} \rangle$
Output: $L^* = \langle \mathcal{T}^*, \mathcal{D}^* \rangle$ and an exceptionality ranking $\mathcal{E}$

$\mathcal{T}^* := \mathcal{T}$;
$\mathcal{D}^* := \mathcal{D}$;
repeat
  $i := 0$;
  $\mathcal{E}_0 := \mathcal{D}^*$;
  $\mathcal{E}_1 := \text{Exceptional}(\langle \mathcal{T}^*, \mathcal{E}_0 \rangle)$;
  while $\mathcal{E}_{i+1} \neq \mathcal{E}_i$ **do**
    $i := i + 1$;
    $\mathcal{E}_{i+1} := \text{Exceptional}(\langle \mathcal{T}^*, \mathcal{E}_i \rangle)$;
  $\mathcal{D}^*_\infty := \mathcal{E}_i$;
  $\mathcal{T}^* := \mathcal{T}^* \cup \{C \sqsubseteq D \mid C \sqsubset\!\!\!\approx D \in \mathcal{D}^*_\infty\}$;
  $\mathcal{D}^* := \mathcal{D}^* \setminus \mathcal{D}^*_\infty$;
until $\mathcal{D}^*_\infty = \emptyset$;
$\mathcal{E} := (\mathcal{E}_0, \ldots, \mathcal{E}_{i-1})$;
return $(L^* = \langle \mathcal{T}^*, \mathcal{D}^* \rangle, \mathcal{E})$;

The ranking of the defeasible axioms is done via the `ComputeRanking`$(\cdot)$ procedure.

In short, the `ComputeRanking`$(\cdot)$ takes as input $L = \langle \mathcal{T}, \mathcal{D} \rangle$ and gives back a new semantically equivalent knowledge base $L = \langle \mathcal{T}^*, \mathcal{D}^* \rangle$ (with $\mathcal{T} \subseteq \mathcal{T}^*$ and $\mathcal{D}^* \subseteq \mathcal{D}$), where possibly some defeasible information in $\mathcal{D}$ has been identified as strict and added to $\mathcal{T}$. Also, a sequence of $\subseteq$-ordered subsets of $\mathcal{D}$ $(\mathcal{E}_0, \ldots, \mathcal{E}_{i-1})$, with increasing level of specificity. That is, in case of potential conflicts, the axioms in a set $\mathcal{E}_j$, $j \geq 0$, have the priority over the axioms in any $\mathcal{E}_i$, $0 \leq i < j$. Now, by considering the ranking $\mathcal{E}_0, \ldots, \mathcal{E}_{i-1}$, we can define a ranking function r that associates to every defeasible concept inclusion in $\mathcal{D}$ a number, representing its level of exceptionality: that is,

$$r(C \sqsubset\!\!\!\approx D) = \begin{cases} j & \text{if } C \sqsubset\!\!\!\approx D \in \mathcal{E}_j \text{ and } C \sqsubset\!\!\!\approx D \notin \mathcal{E}_{j+1} \\ \infty & \text{if } C \sqsubset\!\!\!\approx D \in \mathcal{E}_j \text{ for every } j\,. \end{cases}$$

Similarly, we may associate a rank to a concept C in the following way: consider the result $(L^* = \langle \mathcal{T}^*, \mathcal{D}^* \rangle, \mathcal{E} = (\mathcal{E}_0, \ldots, \mathcal{E}_n))$ of `ComputeRanking`$(\cdot)$. Then

$$r(C) = \begin{cases} j & \text{if } \mathcal{T}^* \models \bigsqcap \overline{\mathcal{E}}_j \sqsubseteq \neg C \text{ and } \mathcal{T}^* \not\models \bigsqcap \overline{\mathcal{E}}_{j+1} \sqsubseteq \neg C \\ \infty & \text{if } \mathcal{T}^* \models \bigsqcap \overline{\mathcal{E}}_j \sqsubseteq \neg C \text{ for every } j\ . \end{cases}$$

Note that $r(C \mathrel{\underline{\sqsubset}\!\!\sim} D) = r(C)$. Now, we will say that $C \mathrel{\underline{\sqsubset}\!\!\sim} D$ is *entailed* by the rational closure of a DL knowledge base L (denoted $L \vdash_{\mathsf{rat}} C \mathrel{\underline{\sqsubset}\!\!\sim} D$) iff $r(C) < r(C \sqcap \neg D)$ [23, Theorem 5.17]). Finally, the procedure `RationalClosure`$(\cdot)$ determines whether $L \vdash_{\mathsf{rat}} C \mathrel{\underline{\sqsubset}\!\!\sim} D$. We recall that the defined entailment relation is indeed as so-called *rational consequence relation* [23], *i.e.* satisfies the following properties:

$$\text{(REF)}\quad L \vdash_{\mathsf{rat}} C \mathrel{\underline{\sqsubset}\!\!\sim} C \qquad \text{Reflexivity}$$

$$\text{(LLE)}\quad \frac{L \vdash_{\mathsf{rat}} C \mathrel{\underline{\sqsubset}\!\!\sim} F \quad L \models C = D}{L \vdash_{\mathsf{rat}} D \mathrel{\underline{\sqsubset}\!\!\sim} F} \qquad \text{Left Logical Equival}$$

$$\text{(RW)}\quad \frac{L \vdash_{\mathsf{rat}} C \mathrel{\underline{\sqsubset}\!\!\sim} D \quad L \models D \sqsubseteq F}{L \vdash_{\mathsf{rat}} C \mathrel{\underline{\sqsubset}\!\!\sim} F} \qquad \text{Right Weakening}$$

$$\text{(CT)}\quad \frac{L \vdash_{\mathsf{rat}} C \mathrel{\underline{\sqsubset}\!\!\sim} D \quad L \models C \sqcap D \mathrel{\underline{\sqsubset}\!\!\sim} F}{L \vdash_{\mathsf{rat}} C \mathrel{\underline{\sqsubset}\!\!\sim} F} \qquad \text{Cut (Cumulative Trans.)}$$

$$\text{(OR)}\quad \frac{L \vdash_{\mathsf{rat}} C \mathrel{\underline{\sqsubset}\!\!\sim} F \quad L \vdash_{\mathsf{rat}} D \mathrel{\underline{\sqsubset}\!\!\sim} F}{L \vdash_{\mathsf{rat}} C \sqcup D \mathrel{\underline{\sqsubset}\!\!\sim} F} \qquad \text{Left Disjunction}$$

$$\text{(RM)}\quad \frac{L \vdash_{\mathsf{rat}} C \mathrel{\underline{\sqsubset}\!\!\sim} F \quad L \not\vdash_{\mathsf{rat}} C \mathrel{\underline{\sqsubset}\!\!\sim} \neg D}{L \vdash_{\mathsf{rat}} C \sqcap D \mathrel{\underline{\sqsubset}\!\!\sim} F} \qquad \text{Rational Monotony}$$

Procedure RationalClosure(L, α)

Input: $L = \langle \mathcal{T}, \mathcal{D} \rangle$ and a query $\alpha = C \mathrel{\underline{\sqsubset}\!\!\sim} D$.
Output: `true` if $L \vdash_{\mathsf{rat}} C \mathrel{\underline{\sqsubset}\!\!\sim} D$, `false` otherwise

```
(L* = ⟨T*, D*⟩, E = (E_0, ..., E_n)) := ComputeRanking(L);
i := 0;
while T* ⊨ ⨅ Ē_i ⊓ C ⊑ ⊥ and i ≤ n do
    i := i + 1;
if i ≤ n then
    return T* ⊨ ⨅ Ē_i ⊓ C ⊑ D;
else
    return T* ⊨ C ⊑ D;
```

We refer the reader to [3] for further explanations and details and limit our presentation to a concluding example.

Example 3. Assume a DL knowledge base $\langle \mathcal{T}, \mathcal{D} \rangle$ with

$$\mathcal{T} = \{Cat \sqsubseteq Feline, Tiger \sqsubseteq Feline, Tiger \sqsubseteq Big, BigFeline = Feline \sqcap Big\ \}$$
$$\mathcal{D} = \{Feline \mathrel{\underline{\sqsubset}\!\!\sim} Agile, Feline \mathrel{\underline{\sqsubset}\!\!\sim} Docile, BigFeline \mathrel{\underline{\sqsubset}\!\!\sim} \neg Docile\ \}\ .$$

By applying the ranking procedure, we end up with

$$r(Cat) = r(Feline) = 0$$
$$r(Feline \sqsubseteq\!\!\!\sim Agile) = r(Feline \sqsubseteq\!\!\!\sim Docile) = 0$$

$$r(Tiger) = r(Feline \sqcap Big) = 1$$
$$r(BigFeline \sqsubseteq\!\!\!\sim \neg Docile) = 1\ .$$

So, for instance, we can conclude that

$$\mathcal{K} \vdash_{\mathsf{rat}} Cat \sqsubseteq\!\!\!\sim Docile\ , \mathcal{K} \vdash_{\mathsf{rat}} Cat \sqsubseteq\!\!\!\sim Agile\ , \mathcal{K} \vdash_{\mathsf{rat}} Cat \sqsubseteq\!\!\!\sim \neg Big$$
$$\mathcal{K} \vdash_{\mathsf{rat}} Tiger \sqsubseteq\!\!\!\sim \neg Docile\ , \mathcal{K} \vdash_{\mathsf{rat}} Cat \sqsubseteq\!\!\!\sim \neg Tiger\ .$$

3 Rational Entailment for DLs via Dl-Programs

In this section we show that, starting from a non-monotonic DL ($\mathcal{SROIQ}$) knowledge base $L = \langle \mathcal{T}, \mathcal{R}, \mathcal{D} \rangle$, we can compile L into a dl-program $\mathcal{K} = \langle \langle \mathcal{T}^*, \mathcal{R} \rangle, P \rangle$ such that the conditions for rational consequence relations are preserved. So, consider a defeasible $\mathcal{SROIQ}$ knowledge base $L = \langle \mathcal{T}, \mathcal{D} \rangle$. Our approach consists of two steps: a ranking step and a compilation step.

Ranking Step. To L we apply the procedure `ComputeRanking`(L) described in Sect. 2.2[3] and, thus, we end up with a new defeasible DL knowledge base $L^* = \langle \mathcal{T}^*, \mathcal{R}, \mathcal{D}^* \rangle$ that correctly separates the strict and the defeasible information contained in the original pair $L = \langle \mathcal{T}, \mathcal{R}, \mathcal{D} \rangle$, and a ranking value $r(C \sqsubseteq\!\!\!\sim D)$ for every defeasible axiom $C \sqsubseteq\!\!\!\sim D \in \mathcal{D}^*$.

Note that in order to adapt the procedures `Exceptional`$(\cdot)$ and `ComputeRanking`$(\cdot)$ to $\mathcal{SROIQ}$ it is sufficient to consider also $\mathcal{R}$ into the ranking procedure: the inputs of both the procedures is a DL knowledge base $L = \langle \mathcal{T}, \mathcal{R}, \mathcal{D} \rangle$ instead of $L = \langle \mathcal{T}, \mathcal{D} \rangle$, and line 3 in Procedure `Exceptional`$(\cdot)$ is modified from $\mathcal{T} \models \bigsqcap \overline{\mathcal{D}} \sqsubseteq \neg C$ to $\mathcal{T} \cup \mathcal{R} \models \bigsqcap \overline{\mathcal{D}} \sqsubseteq \neg C$. The set $\mathcal{R}$ comes out untouched from the ranking procedure, since $\mathcal{D}$ is the only ranked set, and the only possible new strict information is of the form $C \sqsubseteq D$, with C and D concepts, hence it can affect only the content of $\mathcal{T}$. Hence, starting from a knowledge base $\langle \mathcal{T}, \mathcal{R}, \mathcal{D} \rangle$ we end up with a ranked knowledge base $\langle \mathcal{T}^*, \mathcal{R}, \mathcal{D}^* \rangle$.

Dl-program Compilation Step. Given $L^* = \langle \mathcal{T}^*, \mathcal{R}, \mathcal{D}^* \rangle$ from the ranking step, we now compile the defeasible information in $\mathcal{D}^*$ into a a set of dl-rules P, which together with $\mathcal{T}^*, \mathcal{R}$ defines then the final dl-program $\mathcal{K} = \langle \langle \mathcal{T}^*, \mathcal{R} \rangle, P \rangle$.

To alleviate the reading, let $L = \langle \mathcal{T}, \mathcal{R}, \mathcal{D} \rangle := \langle \mathcal{T}^*, \mathcal{R}, \mathcal{D}^* \rangle$; that is, we assume that $\langle \mathcal{T}, \mathcal{R}, \mathcal{D} \rangle$ has already been ranked via the previous ranking step. Now, define a signature $\Phi = \langle \mathtt{P}, \mathtt{C} \rangle$ with $\mathtt{C} = \mathcal{O}$, while $\mathtt{P}$ is composed of the predicates $(c, d, e, \dots)$ representing at the level of programs the concept names in $\mathcal{T} \cup \mathcal{D}$, *i.e.* for each concept

[3] Of course, `Exceptional`$(\cdot)$ and, thus, `ComputeRanking`$(\cdot)$, can be applied to a DL $\mathcal{SROIQ}$ knowledge base L as the classical entailment relation for $\mathcal{SROIQ}$ is decidable.

C in $\mathcal{A}t$ we have an unary predicate c representing it in the rules. We will use the same name with or without the uppercase initial letter to indicate if it is a concept in DL (*e.g.*, $Male$) or a predicate in P (*e.g.*, $male$), respectively. Let us also recall that for each $C \sqsubset\!\!\sim D \in \mathcal{D}$, C and D are either atomic concepts or their negation. Given the ranking of the defeasible axioms in $\mathcal{D}$, let

$$\mathcal{D}_k = \{C \sqsubset\!\!\sim D \mid C \sqsubset\!\!\sim D \in \mathcal{D} \text{ and } r(C \sqsubset\!\!\sim D) = k\}$$

be the subset of $\mathcal{D}$ composed of the axioms with rank value k. Now, define the set

$$\mathfrak{A}_{\mathcal{D}_k} = \{C \mid C \sqsubset\!\!\sim D \in \mathcal{D}_k\}$$

as the set of all the antecedents of the defeasible axioms of rank k. Moreover, we consider also the set of the consequents of the defeasible axioms

$$\mathfrak{C}_{\mathcal{D}} = \{D \mid C \sqsubset\!\!\sim D \in \mathcal{D}\} .$$

Now, for every axiom $C \sqsubset\!\!\sim D \in \mathcal{D}$ of rank k, we create a pair of rules of the form[4]

$$\begin{aligned} d(x) \leftarrow\ & DL[\lambda; C](x), \\ & not\ DL[\lambda; \bigsqcup\{C' \mid C' \in \mathfrak{A}_{\mathcal{D}_m},\ \text{with } m > k\}](x), \\ & not\ \neg d(x) \\ \neg d(x) \leftarrow\ & DL[\lambda; \neg D](x) . \end{aligned} \tag{2}$$

Additionally, for all $C \in \mathfrak{A}_{\mathcal{D}_m}$ with $m > 1$, we also consider a rule

$$\neg c(x) \leftarrow not\ DL[\lambda; C](x) . \tag{3}$$

In all rules above, $\lambda = \{E \uplus e, \neg E \uplus \neg e \mid E \in \mathfrak{C}_{\mathcal{D}}\}$. Note that the size of the grounding of the compiled dl-program is polynomially bounded by the size of the defeasible DL knowledge base.

The intuitive meaning of the rule (2) is the following: assume we have an individual a that is an instance of concept C, which is the antecedent of the defeasible axiom $C \sqsubset\!\!\sim D$ of rank k; if a is not an instance of any other $\mathcal{D}$-antecedent that is more exceptional than C, *i.e.* $not\ DL[\lambda; \bigsqcup\{C' \mid C' \in \mathfrak{A}_{\mathcal{D}_m},\ \text{with } m > k\}](x)$ holds, and $d(a)$ is consistent with our knowledge base, then we can conclude $d(a)$.

On the other hand, the purpose of rule (2) is to update P, in case we derive in L that the conclusion of a defeasible axiom is negated and, thus, the defeasible axiom cannot be applied. λ is necessary to update the DL-base L with the conclusions drawn at the program level.

Finally, rules of form (3) impose that the individuals we are dealing with are as typical as possible. That is, if we are not aware that an exceptional premise applies to them (any concept in $\mathfrak{A}_{\mathcal{D}_m}$, with $m > 1$), then we assume that it doesn't apply (*e.g.*, if we note that a is a bird, but we are not aware that it is a penguin, then we presume that it is not a penguin). In the following, we illustrate our technique via an example.

[4] We assume to simplify double negation: that is, for a concept name F, $\neg\neg F$ is F, and similarly, for a logic program predicate f, $\neg\neg f$ is f. See also Example 4 later on.

Example 4. Assume we have a DL vocabulary with $\mathcal{A}t = \{B, P, F, I, Fi, W, Preyins, Preyfish\}$, $\mathcal{S} = \{Prey\}$, and $\mathcal{O} = \{a, b\}$, were the symbols stand for; $B \mapsto$ *bird*, $P \mapsto$ *penguin*, $F \mapsto$ *flies*, $I \mapsto$ *insect*, $Fi \mapsto$ *fish*, $W \mapsto$ *has wings*, $Preyins \mapsto$ *eats insects*, $Preyfish \mapsto$ *eats fishes*, while $Prey$ is the relation *preys on*.

The DL base $L = \langle \mathcal{T}, \mathcal{D} \rangle$ is composed of

$$\begin{aligned}\mathcal{T} \ = \{ \ & \{a\} \sqsubseteq B, \{b\} \sqsubseteq P, P \sqsubseteq B, I \sqsubseteq \neg Fi, \\ & Preyins = \forall Prey.I \sqcap \exists Prey.\top, \\ & Preyfish = \forall Prey.Fi \sqcap \exists Prey.\top \ \}\end{aligned}$$

$$\mathcal{D} \ = \{B \overset{\sqsubseteq}{\approx} F, P \overset{\sqsubseteq}{\approx} \neg F, B \overset{\sqsubseteq}{\approx} Preyins, P \overset{\sqsubseteq}{\approx} Preyfish, B \overset{\sqsubseteq}{\approx} W \ \} .$$

Now, it can be verified that the ranking step returns the following ranking of axioms $\mathcal{D}$:

$$\begin{aligned}\mathcal{D}_0 \ &= \{B \overset{\sqsubseteq}{\approx} F, B \overset{\sqsubseteq}{\approx} Preyins, B \overset{\sqsubseteq}{\approx} W \ \} \\ \mathcal{D}_1 \ &= \{P \overset{\sqsubseteq}{\approx} \neg F, P \overset{\sqsubseteq}{\approx} Preyfish \ \} .\end{aligned}$$

Therefore, $\mathfrak{A}_{\mathcal{D}_0} = \{B \ \}$, $\mathfrak{A}_{\mathcal{D}_1} = \{P \ \}$, $\mathfrak{C}_{\mathcal{D}} = \{F, \neg F, Preyins, Preyfish, W \ \}$. The compilation step proceeds now as follows. We define a vocabulary $\Phi = \langle \mathtt{P}, \mathtt{C} \rangle$ with $\mathtt{C} = \{a, b\}$, while $\mathtt{P}$ is composed of predicates that represent at the program level the DL atomic concepts and roles: that is, $\mathtt{P} = \{b, p, f, i, fi, w, preyins, preyfish, prey\}$. The program P, resulting from the compilation step, is composed of the following rules:

$$\begin{aligned} f(x) &\leftarrow DL[\lambda; B](x), not\ DL[\lambda; P](x), not\ \neg f(x) \\ \neg f(x) &\leftarrow DL[\lambda; \neg F](x) \end{aligned}$$

$$\begin{aligned} preyins(x) &\leftarrow DL[\lambda; B](x), not\ DL[\lambda; P](x), not\ \neg preyins(x) \\ \neg preyins(x) &\leftarrow DL[\lambda; \neg Preyins](x) \end{aligned}$$

$$\begin{aligned} w(x) &\leftarrow DL[\lambda; B](x), not\ DL[\lambda; P](x), not\ \neg w(x) \\ \neg w(x) &\leftarrow DL[\lambda; \neg W](x) \end{aligned}$$

$$\begin{aligned} \neg f(x) &\leftarrow DL[\lambda; P](x), not\ f(x) \\ f(x) &\leftarrow DL[\lambda; F](x) \end{aligned}$$

$$\begin{aligned} preyfish(x) &\leftarrow DL[\lambda; P](x), not\ \neg preyfish(x) \\ \neg preyfish(x) &\leftarrow DL[\lambda; \neg Preyfish](x) \end{aligned}$$

$$\neg p(x) \leftarrow not\ DL[\lambda; P](x) \ ,$$

with

$$\begin{aligned}\lambda = \ \{ \ & F \uplus f, \neg F \uplus \neg f, W \uplus w, \neg W \uplus \neg w, Preyins \uplus preyins, \\ & \neg Preyins \uplus \neg preyins, Preyfish \uplus preyfish, \neg Preyfish \uplus \neg preyfish \ \} .\end{aligned}$$

Now, note that the only answer set to the program P is the interpretation

$$I = \{f(a), preyins(a), w(a), \neg p(a), \neg f(b), preyfish(b)\} .$$

In fact, I is the least model of the grounded positive program P^I

$$\begin{aligned} f(a) &\leftarrow DL[\lambda; B](a) \\ preyins(a) &\leftarrow DL[\lambda; B](a) \\ w(a) &\leftarrow DL[\lambda; B](a) \\ \neg f(b) &\leftarrow DL[\lambda; P](b) \\ preyfish(b) &\leftarrow DL[\lambda; P](b) \\ \neg p(a) &\leftarrow \ . \end{aligned}$$

So, we obtain the intuitive conclusions that, if we are aware about an individual that it is just a bird, we can conclude that, presumably, it flies, eats insects and has wings. On the other hand, if we are informed that it is a penguin, we can conclude that it doesn't fly and eats fishes.

As well known and already noted in [8], having *nominal concepts* may end up in having multiple extensions, *i.e.*, in our context, we may have multiple strong answer sets as shown with the following simple example.

Example 5. Consider a knowledge base $L = \langle \mathcal{T}, \mathcal{D} \rangle$, with

$$\begin{aligned} \mathcal{T} &= \{\{a\} \sqsubseteq \exists R.\{b\}, C = D \sqcap \forall R.\neg D \ \} \\ \mathcal{D} &= \{\top \tilde{\sqsubseteq} C \ \} . \end{aligned}$$

By applying our method we obtain the following program P

$$\begin{aligned} c(x) &\leftarrow DL[\lambda; \top](x), not\ \neg c(x) \\ \neg c(x) &\leftarrow DL[\lambda; \neg C](x) . \end{aligned}$$

Now, it can be verified that from the dl-program $\mathcal{K} = \langle L, P \rangle$ we obtain now two strong answer sets: namely,

$$\begin{aligned} I &= \{c(a), \neg c(b) \ \} \\ I' &= \{c(b), \neg c(a) \ \} . \end{aligned}$$

Nevertheless, the main result of this paper is that each strong answer set defines a rational consequence relation. In fact, we consider the content of the DL base updated with the content of an answer set I by means of the operator $\uplus$. That is, we define a consequence relation $\models_{P^I}$ where, $\mathcal{K} = \langle L, P \rangle \models_{P^I} C(a)$ iff the DL base L augmented, using $\uplus$, with the content of a strong answer set I of $\mathcal{K}$, entails $C(a)$. Specifically, we can show that

Proposition 1. *Given $\mathcal{K} = \langle L, P \rangle$, were L contains a $\mathcal{SROIQ}$ TBox and a $\mathcal{SROIQ}$ RBox, P is the result of compiling L into dl-rules, and a strong answer set I of $\mathcal{K}$. Then the consequence relation $\models_{P^I}$ satisfies the following properties:*[5]

[5] For ease of comprehension, we write concept assertions as $D(b)$ in place of the equivalent inclusion axiom $\{b\} \sqsubseteq D$ in expressions like $L \cup \{D(b)\}$.

$$\mathrm{REF}_{\mathrm{DL}} \quad \langle L, P\rangle \models_{P^I} C(a) \text{ for every } C(a) \in L$$

$$\mathrm{LLE}_{\mathrm{DL}} \quad \frac{\langle L \cup \{D(b)\}, P\rangle \models_{P^I} C(a) \;\; L \models D = E}{\langle L \cup \{E(b)\}, P\rangle \models_{P^I} C(a)}$$

$$\mathrm{RW}_{\mathrm{DL}} \quad \frac{\langle L, P\rangle \models_{P^I} C(a) \;\; L \models C \sqsubseteq D}{\langle L, P\rangle \models_{P^I} D(a)}$$

$$\mathrm{CT}_{\mathrm{DL}} \quad \frac{\langle L \cup \{D(b)\}, P\rangle \models_{P^I} C(a) \;\; \langle L, P\rangle \models_{P^I} D(b)}{\langle L, P\rangle \models_{P^I} C(a)}$$

$$\mathrm{OR}_{\mathrm{DL}} \quad \frac{\langle L \cup \{D(b)\}, P\rangle \models_{P^I} C(a) \;\; \langle L \cup \{E(b)\}, P\rangle \models_{P^I} C(a)}{\langle L \cup \{(D \sqcup E)(b)\}, P\rangle \models_{P^I} C(a)}$$

$$\mathrm{RM}_{\mathrm{DL}} \quad \frac{\langle L, P\rangle \models_{P^I} C(a) \;\; \langle L, P\rangle \not\models_{P^I} \neg D(b)}{\langle L \cup \{D(b)\}, P\rangle \models_{P^I} C(a)}$$

Proof. (Sketch) The proofs for $\mathrm{REF}_{\mathrm{DL}}$, $\mathrm{LLE}_{\mathrm{DL}}$ are $\mathrm{RW}_{\mathrm{DL}}$ are straightforward, considering the set-theoretic semantics of DLs.

In what follows, given the strong answer set I, the expression I^{DL} indicates the obvious translation of the answer set into the DL base L, so that $\langle L, P\rangle \models_{P^I} C(a)$ iff $L \cup I^{DL} \models C(a)$.

For $\mathrm{CT}_{\mathrm{DL}}$, if I is an answer set for both $\langle L, P\rangle$ and $\langle L \cup \{D(b)\}, P\rangle$, then we have $L \cup \{D(b)\} \cup I^{DL} \models C(a)$ and $L \cup I^{DL} \models D(b)$, and, since every classical DL consequence relation $\models$ satisfies CT, we have $L \cup I^{DL} \models C(a)$, *i.e.* $\langle L, P\rangle \models_{P^I} C(a)$.

For $\mathrm{OR}_{\mathrm{DL}}$, if I is an answer set for both $\langle L \cup \{D(b)\}, P\rangle$ and $\langle L \cup \{E(b)\}, P\rangle$, it must be an answer set also for $\langle L \cup \{(D \sqcup E)(b)\}, P\rangle$: since $\models$ is monotonic, it is not possible to derive from $L \cup \{(D \sqcup E)(b)\}$ some element of the set $B^-(r)$ of some r in P that could not be derived from $L \cup \{D(b)\}$ or $L \cup \{E(b)\}$; hence a rule can be eliminated from P only if also $L \cup \{D(b)\}$ or $L \cup \{E(b)\}$ would eliminate it. Given the validity of OR for $\models$, we have that $L \cup \{D(b)\} \cup I^{DL} \models C(a)$ and $L \cup \{E(b)\} \cup I^{DL} \models C(a)$ imply $L \cup \{(D \sqcup E)(b)\} \cup I^{DL} \models C(a)$, *i.e.* $\langle L \cup \{D(b)\}, P\rangle \models_{P^I} C(a)$.

For $\mathrm{RM}_{\mathrm{DL}}$, assume $\langle L, P\rangle \models_{P^I} C(a)$ and $\langle L, P\rangle \not\models_{P^I} \neg D(b)$. It is sufficient to show that the answer set I must be an answer set also for $\langle L \cup \{D(b)\}, P\rangle$. Assume the opposite, *i.e.* I is not an answer set for $\langle L \cup \{D(b)\}, P\rangle$. Then, there must be in P a rule r associated to a defeasible axiom with rank equal to k s.t. $not\ \alpha \in B^-(r)$, where α is some literal s.t. $L \not\models \alpha^{DL}$ and $L \cup \{D(b)\} \models \alpha^{DL}$ (α^{DL} is the translation of α into the DL-language). In such a case, r must have been a ground rule of form

$$e(c) \leftarrow DL[\lambda; C](c), not\ DL[\lambda; \bigsqcup\{C' \mid C' \in \mathfrak{A}_{\mathcal{D}_m}, \text{ with } m > k\}](c), not \neg e(c)\ .$$

α cannot be $\neg e(c)$, since from the activation of the rule we would have $\langle L, P\rangle \models_{P^I} e(c)^{DL}$, and consequently $\langle L, P\rangle \models_{P^I} \neg D(b)$, which contradicts the hypothesis. As a consequence, α must be the dl-atom of the form $not\ DL[\lambda; \bigsqcup\{C'\}](c)$. But then again, the activation of the rule for the individual c under $\langle L, P\rangle$ implies that the individual c is ranked at the value k (the rank of an individual a is the rank of $\{a\}$, *i.e.* $r(\{a\})$).

Having every C' a higher ranking value than k, and so also $\bigsqcup\{C'\}$, we can conclude $\langle L, P\rangle \models_{P^I} \neg \bigsqcup\{C'\}(c)$, from which, again, we have $\langle L, P\rangle \models_{P^I} \neg D(b)$, contrary to hypothesis. This concludes the proof.

4 Related Work

Several non-monotonic DLs exist, but somewhat related to our proposal are [2–6,8,9,11,12,15–18,24,25], as they address the application of the preferential semantics [23]. As far as we know, [2,5,6] are the only works that consider also a DL as expressive as $\mathcal{SROIQ}$. [5,6] propose a language, associated to a preferential semantics, that is more expressive than the one presented here, allowing the representation of many forms of defeasibily. However, at the moment such a logic is still missing a mature entailment relation. Bonatti [2] defines a semantic construction that extends rational closure to $\mathcal{SROIQ}$: the previous proposals [3,12,18] rely on the *disjoint model union property*, that does not hold for a DL as expressive as $\mathcal{SROIQ}$, while Bonatti proposes an alternative construction based on *stable rankings*, that is applicable for every DLs. We are not aware of any approach that relies on dl-programs, but [15,16] propose an ASP-based decision procedure for the DL $\mathcal{SROEL}$, relying on a Datalog encoding of the DL knowledge base.

5 Conclusions

The introduction of rational monotonicity into the field of dl-programs allows the use of a non-monotonic formalism that at the same time satisfies important logical properties and gives back intuitive conclusions. From the implementation point of view, our proposal allows to compile the decision procedures into dl-programs and, thus, it can be implemented on top of existing reasoners supporting dl-programs such as DLV.

Regarding future work, we believe that two aspects are particularly urgent. Firstly, a comparison with the semantic characterisation of rational closure for $\mathcal{SROIQ}$ in [2].

Also, we would like to address the computational complexity of our approach. So far, we know that computing the rankings can be done in polynomial number of calls [8, 12] to an oracle deciding $\mathcal{SROIQ}$ entailment (the latter is complete for 2NEXP [21]). It remains to be seen whether, by reasoning similarly as done in [13], in which it has been shown that w.r.t. $\mathcal{SHOIN}$ the existence of answer sets, cautious and brave reasoning problems are complete for $\mathrm{P}^{\mathrm{NEXP}}$ (recall that the entailment problem for $\mathcal{SHOIN}$ is complete for NEXP [26]), the same problems are complete for $\mathrm{P}^{\mathrm{2NEXP}}$ w.r.t. our $\mathcal{SROIQ}$ setting, *i.e.* solvable in polynomial time by relying on an oracle for 2NEXP.

Eventually, from the inferential point of view rational closure has some well-known weaknesses: while there can be intuitive, desirable conclusions that cannot be derived [22], it remains an important basic construction that can be extended into richer entailment relations such as those proposed in [7,10,11,17]. Future work will be partly dedicated to extending the present method to some of these entailment relations.

Acknowledgments. This research was supported by TAILOR, a project funded by EU Horizon 2020 research and innovation programme under GA No. 952215.

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.: The Description Logic Handbook. Cambridge University Press, Cambridge (2003)

2. Bonatti, P.A.: Rational closure for all description logics. Artif. Intell. **274**, 197–223 (2019)
3. Britz, K., Casini, G., Meyer, T., Moodley, K., Sattler, U., Varzinczak, I.: Principles of KLM-style defeasible description logics. ACM Trans. Comput. Log. **22**(1), 1:1–1:46 (2021)
4. Britz, K., Meyer, T., Varzinczak, I.J.: Concept model semantics for dl preferential reasoning. In: DL 2011 (2011)
5. Britz, K., Varzinczak, I.: Context-based defeasible subsumption for dSROIQ. In: COMMONSENSE 2017. CEUR Workshop Proceedings, vol. 2052. CEUR-WS.org (2017)
6. Britz, K., Varzinczak, I.: Towards defeasible SROIQ. In: DL-17. CEUR, vol. 1879 (2017)
7. Casini, G., Meyer, T., Moodley, K., Nortjé, R.: Relevant closure: a new form of defeasible reasoning for description logics. In: Fermé, E., Leite, J. (eds.) JELIA 2014. LNCS (LNAI), vol. 8761, pp. 92–106. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11558-0_7
8. Casini, G., Straccia, U.: Rational closure for defeasible description logics. In: JELIA-10, pp. 77–90. No. 6341 in LNAI, Springer (2010)
9. Casini, G., Straccia, U.: Defeasible inheritance-based description logics. In: IJCAI 2011, pp. 813–818 (2011)
10. Casini, G., Straccia, U.: Lexicographic closure for defeasible description logics. In: Proceedings of the Eighth Australasian Ontology Workshop - AOW 2012, pp. 28–39. No. 969 in CEUR Workshop Proceedings, CEUR (2012)
11. Casini, G., Straccia, U.: Defeasible inheritance-based description logics. J. Artif. Intell. Res. **48**, 415–473 (2013). https://doi.org/10.1613/jair.4062
12. Casini, G., Straccia, U., Meyer, T.: A polynomial time subsumption algorithm for nominal safe $\mathcal{ELO}_{\bot}$ under rational closure. Inf. Sci. **501**, 588–620 (2019)
13. Eiter, T., Ianni, G., Lukasiewicz, T., Schindlauer, R., Tompits, H.: Combining answer set programming with description logics for the semantic web. Artif. Intell. **172**(12–13), 1495–1539 (2008)
14. Gelfond, M., Lifschitz, V.: Classical negation in logic programs and disjunctive databases. New Gen. Comput. **9**, 365–385 (1991)
15. Giordano, L., Dupré, D.T.: ASP for minimal entailment in a rational extension of SROEL. Theory Pract. Log. Program. **16**(5–6), 738–754 (2016)
16. Giordano, L., Dupré, D.T.: Reasoning in a rational extension of SROEL. In: DL 2016. CEUR Workshop Proceedings, vol. 1577. CEUR-WS.org (2016)
17. Giordano, L., Gliozzi, V.: A reconstruction of multipreference closure. Artif. Intell. **290** (2021)
18. Giordano, L., Gliozzi, V., Olivetti, N., Pozzato, G.: Semantic characterization of rational closure: From propositional logic to description logics. Artif. Intell. **226**, 1–33 (2015)
19. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible SROIQ. In: KR-06, pp. 57–67. AAAI Press, Palo Alto (2006)
20. Horrocks, I., Sattler, U.: Decidability of $\mathcal{SHIQ}$ with complex role inclusion axioms. Art. Intell. **160**, 79–104 (2004)
21. Kazakov, Y.: $\mathcal{RIQ}$ and $\mathcal{SROIQ}$ are harder than $\mathcal{SHOIQ}$. In: KR-08, pp. 274–284 (2008)
22. Lehmann, D.: Another perspective on default reasoning. Ann. Math. Art. Int. **15**, 61–82 (1995)
23. Lehmann, D., Magidor, M.: What does a conditional knowledge base entail? Artif. Intell. **55**(1), 1–60 (1992). https://doi.org/10.1016/0004-3702(92)90041-U
24. Pensel, M., Turhan, A.: Reasoning in the defeasible description logic $\mathcal{EL}_{\bot}$. Int. J. Appr. Reas. **103**, 28–70 (2018)
25. Straccia, U.: Default inheritance reasoning in hybrid kl-one-style logics. IJCAI-93, pp. 676–681 (1993)
26. Tobies, S.: Complexity results and practical algorithms for logics in knowledge representation. Ph.D. thesis, RWTH-Aachen (2001)