Default theory for Well Founded Semantics with Explicit Negation

Luís Moniz Pereira José Júlio Alferes Joaquim Nunes Aparício
CRIA, Uninova and DCS, U. Nova de Lisboa, 2825 Monte da Caparica, Portugal
{lmp|jja|jna}@fct.unl.pt

Abstract

One aim of this paper is to define a default theory for Well Founded Semantics of logic programs which have been extended with explicit negation, such that the models of a program correspond exactly to the extensions of the default theory corresponding to the program.

To do so we must introduce a new default theory semantics that satisfies principles of modularity, enforcedness, and uniqueness of minimal extension (if it has an extension), which have a natural counterpart in the program semantics. Other default theories, with which we compare our own, do not satisfy all these principles, namely Reiter's default theory, Baral and Subrahmanian's well founded extensions, and Przymusinski's stationary extensions.

The relationship between logic programs and defaults theories opens the way for a mutual fertilization, which we have enhanced:

On the one hand, we widen the class of programs which can be given a semantics by a suitable default theory, show that explicit negation can be translated into classical negation in such a default theory, and that it clarifies the use of rules in extended logic programs.

On the other hand, since our logic program semantics is definable by a monotonic fixpoint operator, it has desirable computational properties, including top–down and bottom–up procedures. As our semantics is sound with respect to Reiter's default semantics, whenever an extension exists, we thus provide sound methods for computing the intersection of all extensions for an important subset of Reiter's default theories.

1 Introduction

A relationship between logic programs and default theories was first proposed in [3] and [4]. The idea is to translate every program rule into a default one and then compare extensions of the default theory with the semantics of the corresponding program.

In [4], stable model semantics [5] was shown equivalent to a special case of default theories in the sense of Reiter [15]. This result was generalized in [6] to programs with explicit negation and answer-set semantics, where they claim that explicit negation corresponds in fact to classical negation used in default theories.

Well Founded Semantics for Default Theories [1] extends Reiter's semantics of default theories, resolving some issues of the latter, namely that some theories have no extension and that some theories have no least extension. Based on the way such issues were resolved in [2], the well founded semantics for programs without explicit negation was shown equivalent to a special case of the extension classes of default theories in the sense of [1]. It turns out that in attempting to extend this result to extended logic programs with explicit negation we get some unintuitive results and no semantics of such logic programs relates to known default theories.

To overcome that, here we identify principles a default theory semantics should enjoy, and introduce a default theory semantics that extends that of [2] to the larger class of logic programs, and complies with those principles.

Such relationship to a larger program class improves the cross–fertilization between logic programs and default theories, since we generalize previous results concerning their relationship [1, 2, 3, 4, 6, 13], and also because there is an increasing use of logic programming with explicit negation for nonmonotonic reasoning [6, 9, 10, 11, 17]. It also clarifies the meaning of logic programs combining both explicit negation and negation by default. In particular, it shows in what way explicit negation corresponds to classical negation in our default theory, and elucidates the use of rules in extended logic programs. Like defaults rules are unidirectional, so their contrapositives are not implicit; the rule connective $\leftarrow$ is not material implication but has rather the flavour of an inference rule, as in defaults.

On the other hand, since our logic program semantics is definable by a monotonic fixpoint operator, it has desirable computational properties, including top–down and bottom–up procedures. As our semantics is sound with respect to Reiter's default semantics, whenever an extension exists, we thus provide sound methods for computing the intersection of all extensions for an important subset of Reiter's default theories .

2 Language Used

Given a first order language *Lang* [12], an extended logic program is a set of rules of the form $H \leftarrow B_1, \ldots, B_n, \sim C_1,\ \ldots,\ \sim C_m \quad m \geq 0, n \geq 0$, where $H, B_1, \ldots, B_n, C_1, \ldots, C_m$ are classical literals. A (syntactically) classical literal (or explicit literal) is either an atom A or its explicit negation $\neg A$. We also use the symbol $\neg$ to denote complementary literals in the sense of explicit negation. Thus $\neg\neg A = A$. The symbol $\sim$ stands for negation by default[1]. $\sim L$ is called a default literal. Literals are either classical or default literals. A set of rules stands for all its ground instances wrt *Lang*.

Concerning default theories, $Lang(AT)$ is the propositional language generated by considering the ground atoms of *Lang* to be propositional symbols, where *Lits* is the set of its ground literals.

Definition 2.1 (Default rule) *A propositional default is a triple* $d = (p(d), j(d), c(d))$ *where* $p(d)$ *and* $c(d)$ *are formulas of* $Lang(AT)$ *and* $j(d)$ *is a finite subset of*

[1] This designation has been used in the literature instead of the more operational *"negation as failure (to prove)"*. Another appropriate designation is *"implicit negation"*, in contradistinction to explicit negation.

Lang(AT). p(d) (resp. j(d), resp. c(d)) is called the prerequisite *(resp.* justification, *resp.* consequence*) of default d. The default d is also denoted* $\frac{p(d):j(d)}{c(d)}$.

Definition 2.2 (Default theory) *A default theory* Δ *is a pair* (D, W) *where* $W \subseteq$ *Lang and D is a set of default rules.*

3 *WFSX* overview

In this section we briefly review $WFSX$ semantics for logic programs extended with explicit negation. For full details the reader is referred to [8].

$WFSX$ follows from one basic "coherence" requirement: $\neg L$ implies $\sim L$ (if L is explicitly false, L must be false) for any explicit literal L.

Example 1 *[8]* Consider program $P = \{a \leftarrow \sim b,\quad b \leftarrow \sim a,\quad \neg a \leftarrow\}$.

If $\neg a$ were to be simply considered as a new atom symbol, say a', and WFS used to define the semantics of P (as suggested in [14]), the result would be $\{\neg a, \sim\neg b\}$, so that $\neg a$ is true and a is undefined. We insist that $\sim a$ should hold, and a not, because $\neg a$ does. Accordingly, the WFSX of P is $\{\neg a, b, \sim a, \sim\neg b\}$, since b follows from $\sim a$.

We begin by providing a definition of interpretation for programs with explicit negation which incorporates coherence from the start.

Definition 3.1 (Interpretation) *[8] By an interpretation I of a language Lang we mean any set* $T \cup \sim F$[2]*, where T and F are disjoint subsets of classical literals over the Herbrand base, and if* $\neg L \in T$ *then* $L \in F$ *(coherence)*[3]*. The set T contains all ground classical literals* true *in I, the set F contains all ground classical literals* false *in I. The truth value of the remaining classical literals is* undefined *(The truth value of a default literal* $\sim L$ *is the 3-valued complement of L.)*

We next extend with an additional rule the P modulo I transformation of [12], itself an extension of the Gelfond-Lifschitz modulo transformation, to account for coherence.

Definition 3.2 ($\frac{P}{I}$ transformation) *[8] Let P be an extended logic program and let I be an interpretation. By* $\frac{P}{I}$ *we mean a program obtained from P by performing the following three operations for every atom A :*

- *Remove from P all rules containing a negative premise* $L = \sim A$ *such that* $A \in I$*.*
- *Remove from P all rules containing a premise L (resp.* $\neg L$*) such that* $\neg L \in I$ *(resp.* $L \in I$*).*
- *Remove from all remaining rules of P their negative premises* $L = \sim A$ *such that* $\sim A \in I$*.*
- *Replace all the remaining negative premises by proposition* $\mathbf{u}$[4]*.*

---

[2] By $\sim\{a_1, \ldots, a_n\}$ we mean $\{\sim a_1, \ldots, \sim a_n\}$.

[3] For any literal L, if L is explicitly false L must be false. Note that the complementary condition "if $L \in T$ then $\neg L \in F$" is implicit.

[4] The special proposition $\mathbf{u}$ is *undefined* in all interpretations.

The resulting program $\frac{P}{I}$ is by definition non-negative, and it always has a unique $least(\frac{P}{I})$, where $least(\frac{P}{I})$ is:

Definition 3.3 (Least-operator) *We define $least(P)$, where P is a non-negative program, as the set of literals $T \cup \sim F$ obtained as follows:*

- *Let P' be the non-negative program obtained by replacing in P every negative classical literal $\neg L$ by a new atomic symbol, say $'\neg\_L'$.*
- *Let $T' \cup \sim F'$ be the least 3-valued model of P'.*
- *$T \cup \sim F$ is obtained from $T' \cup \sim F'$ by reversing the replacements above.*

The least 3-valued model of a non-negative program can be defined as the least fixpoint of the following generalization of the van Emden–Kowalski least model operator Ψ for definite logic programs:

Definition 3.4 Ψ^* *operator*
Suppose that P is a non-negative program, I is an interpretation of P and A is a ground atom. Then $\Psi^*(I)$ is an interpretation defined as follows:

- *$\Psi^*(I)(A) = 1$ iff there is a rule $A \leftarrow A_1, \ldots, A_n$ in P such that $I(A_i) = 1$ for all $i \leq n$.*
- *$\Psi^*(I)(A) = 0$ iff for every rule $A \leftarrow A_1, \ldots, A_n$ there is an $i \leq n$ such that $I(A_i) = 0$.*
- *$\Psi^*(I)(A) = 1/2$, otherwise.*

To avoid incoherence, a partial operator is defined that transforms any non-contradictory set of literals into an interpretation, whenever contradiction[5] is not present.

Definition 3.5 (The Coh operator) *[8] Let $I = T \cup \sim F$ be a set of literals such that T is not contradictory. We define $Coh(I) = I \cup \sim\{\neg L \mid L \in T\}$.*

Definition 3.6 (The Φ operator) *[8] Let P be a logic program and I an interpretation, and let $J = least(\frac{P}{I})$. If $Coh(J)$ exists we define $\Phi_P(I) = Coh(J)$. Otherwise $\Phi_P(I)$ is not defined.*

Definition 3.7 (WFS with explicit negation) *[8] An interpretation I of an extended logic program P is called an Extended Stable Model (XSM) of P iff $\Phi_P(I) = I$. The F-least Extended Stable Model is called the Well Founded Model. The semantics of P is determined by the set of all $XSMs$ of P.*

4 A Default Theory for Extended Logic Programs

The relationship between the semantics of the default theories of [2] and the semantics of the class of logic programs extended with explicit negation has not been defined. We will now introduce a default theory corresponding to this broader class of programs (whose $WFSX$ semantics was reviewed above).

[5] We say a set of literals S is contradictory iff for some literal L, $L \in S$ and $\neg L \in S$.

4.1 Principles Required of Default Theories

Next we argue about principles a default theory semantics should enjoy, and relate it to logic programs extended with explicit negation, where the said principles can also be considered desireable.

Definition 4.1 (Uniqueness of minimal extension) *Whenever a default theory has an extension there is a minimal one.*

It is well known that Reiter's default theories do not comply with this principle.

Definition 4.2 (Union of theories) *By the union of two default theories* $\Delta_1 = (D_1, W_1)$ *and* $\Delta_2 = (D_2, W_2)$ *with languages* $L(\Delta_1)$ *and* $L(\Delta_2)$ *we mean the theory* $\Delta = \Delta_1 \cup \Delta_2 = (D_1 \cup D_2, W_1 \cup W_2)$ *with language* $L(\Delta) = L(\Delta_1) \cup L(\Delta_2)$, *whenever* $W_1 \cup W_2$ *is consistent.*

Example 2 Consider the two default theories:

$$\Delta_1 = (\{\tfrac{:\neg a}{\neg a}, \tfrac{:a}{a}\}, \{\}) \quad \Delta_2 = (\{\tfrac{:b}{b}\}, \{\})$$

Classical default theory, well–founded semantics, and stationary semantics all identify $\{b\}$ as the single extension of Δ_2.

Since the languages of the two theories are disjoint, one would expect their union to include b in all its extensions. However, both the well founded semantics as well as the least stationary semantics give the value unknown to b in the union theory, and therefore are not modular (cumulative). There is an objectionable interaction among the default rules of both theories when put together. Classical default theory is modular but has two extensions: $\{\neg a, b\}$ and $\{a, b\}$, failing to give a unique minimal extension to the union.

Definition 4.3 (Modularity) *Let* Δ_1, Δ_2 *be two default theories such that* $L(\Delta_1) \cap L(\Delta_2) = \{\}$ *and let* $\Delta = \Delta_1 \cup \Delta_2$, *with extensions* $E^i_{\Delta_1}$ $E^j_{\Delta_2}$ *and* E^k_{Δ}. *A semantics for default theories is* modular *iff:*

$$\forall_A (\forall_i A \in E^i_{\Delta_1} \Rightarrow \forall_k A \in E^k_{\Delta})$$
$$\forall_A (\forall_j A \in E^j_{\Delta_2} \Rightarrow \forall_k A \in E^k_{\Delta})$$

Consider now the following examples:

Example 3 Let $(D_1, W_1) = (\{d_1 = \tfrac{:\neg b}{a}, d_2 = \tfrac{:\neg a}{b}\}, \{\})$. The two classical extensions are $\{a\}$ and $\{b\}$. Stationary default semantics has one more extension, namely $\{\}$.

Example 4 Let $(D_2, W_2) = (\{d_1 = \tfrac{:\neg b}{a}, d_2 = \tfrac{:\neg a}{b}\}, \{\neg a\})$. The only classical extension is $\{\neg a, b\}$. In the least stationary extension, $E = \Gamma^2_\Delta(E) = \{\neg a\}$, $j(d_2) \in E$ but $c(d_2) \notin E$.

Definition 4.4 *Given an extension* E:

- *a default* d *is* applicable *in* E *iff* $p(d) \subseteq E$ *and* $\neg j(d) \cap E = \{\}$

- *an applicable default d is* applied in E *iff* $c(d) \in E$

In classical default semantics every applicable default is applied. This prevents the uniqueness of a minimal extension. In example 3, because one default is always applied, one can never have a single minimal extension. In [2, 13], in order to guarantee a unique minimal extension, it becomes possible to apply or not an applicable default. However, this abandons the notion of maximality of application of defaults of classical default theory. But, in example 4, we argue that at least rule d_2 should be applied.

We want to retain the principle of uniqueness of minimal extension coupled with a notion of maximality of application of defaults which we call enforcedness.

Definition 4.5 (Enforcedness) *Given a theory Δ with extension E, a default d is* enforceable in E *iff $p(d) \subseteq E$ and $j(d) \subseteq E$. An extension* is enforced *if all enforceable defaults in D are applied.*

We argue that, whenever E is an extension, if a default is enforceable then it must be applied. Note that an enforceable default is always applicable.

Another way of viewing enforcedness is that if d is an enforceable default, and E is an extension, then the default rule d must be understood as an inference rule $p(d), j(d) \rightarrow c(d)$ and so $c(d) \in E$ must hold.

The well founded semantics and stationary semantics both sanction minimal extensions where enforceable defaults are not applied, viz. example 4. However, in this example they still allow an enforced extension $\{b, \neg a\}$. This is not the case in general:

Example 5 Let $(D, W) = (\{\frac{:\neg b}{c}, \frac{:\neg a}{b}, \frac{:\neg a}{a}\}, \{\neg b\})$. The only stationary extension is $\{\neg b\}$, which is not enforced.

4.2 Ω–Default Theory

In this section we introduce a default theory semantics which is modular and enforced. Moreover, when it is defined it has a unique minimal extension.

When relating non–disjunctive logic programs to defaults it is customary to restrict prerequisites, justifications and conclusions to ground literals. Also, each program rule corresponds to a such default, and the theory W is empty [1, 2, 3, 4, 6, 13]. We shall do the same, though in our case, we also allow W to contain ground literals.

If this may seem an excessive restriction, we prove below that the extensions of such default theories correspond to models of normal logic programs extended with explicit negation, and so have comparable expressibility.

In order to relate default theories to extended logic programs, we must provide a modular semantics for default theories, except if they are contradictory as in the example below:

Example 6 In the default theory $(\{\frac{:}{\neg a}, \frac{:}{a}\}, \{\})$ its two defaults with empty prerequesites and justifications should always be applied, which clearly enforces a contradiction. Note that this would also be the case if the default theory were to be written as $(\{\}, \{a, \neg a\})$.

Consider now example 2, which alerted us about nonmodularity in stationary or well founded default semantics, where $D = \{\frac{:\neg a}{\neg a}, \frac{:a}{a}, \frac{:b}{b}\}$ and $\{\}$ is the least extension. This result obtains because $\Gamma_\Delta(\{\})$, by having a and $\neg a$ forces, via the deductive closure operator Cn, $\neg b$ (and all the other literals) to belong to it. This implies the nonapplicability of the third default in the second iteration. For that not to happen one should inhibit $\neg b$ from belonging to $\Gamma_\Delta(\{\})$, which can be done by preventing the application of Cn. In the language to which we restrict ourselves this is not problematic because, as formulae are literals, Cn does not introduce anything except in the contradictory case. Next we define $\Gamma'_\Delta(E)$, similar to $\Gamma_\Delta(E)$ but where the Cn operator is absent.

Definition 4.6 ($\Gamma'_\Delta(E)$) *Suppose (D,W) is a default theory and E is a* context. *The operator $r^{E,D}(S)$, mapping sets of literals S into sets of literals, is defined as follows:*

$$r^{E,D}(S) = S \cup \{c(d) | d \in D, p(d) \subseteq S, \neg j(d) \cap E = \{\}\} \tag{1}$$

We define the sequence:

$$\begin{array}{l} r_0^{E,D}(W) = W \\ r_{n+1}^{E,D}(W) = r^{E,D}(r_n^{E,D}(W)) \\ r_\infty^{E,D}(W) = \bigcup_{n=0}^{\infty} r_n^{E,D}(W) \end{array}$$

and $\Gamma'_\Delta(E) = r_\infty^{E,D}(W)$.

Reconsider now example 4, that showed that stationary default extensions are not always enforced. The nonenforced extension is (the least extension) $E = \Gamma^2(E) = \{\neg a\}$, where $\Gamma(E) = \{\neg a, a, b\}$. The semantics obtained is that $\neg a$ is true and a is undefined.

To avoid this counterintuitive result we want to ensure that, for an extension $E : \forall d \in D \;\; \neg c(d) \in E \Rightarrow c(d) \notin \Gamma(E)$, i.e. if $\neg c(d)$ is true then $c(d)$ is false[6].

It is easily recognized that this condition is satisfied by seminormal default theories: if $\neg c(d)$ belongs to an extension then any seminormal rule with conclusion $c(d)$ cannot be applied. This principle is exploited in our semantics.

Definition 4.7 *Given a default theory Δ, we dub Δ^s its seminormal version*[7] *obtained as follows: There is a rule $d^s = \frac{p(d):j(d),c(d)}{c(d)}$ in Δ^s for each default rule $d = \frac{p(d):j(d)}{c(d)}$ in Δ.*

Definition 4.8 (Ω–extension) *For a theory Δ we define $\Omega_\Delta(E) = \Gamma'_\Delta(\Gamma'_{\Delta^s}(E))$. E is an Ω–extension of Δ iff:*

- *E contains no pair of complementary literals A and $\neg A$*
- *$E = \Omega_\Delta(E)$*

[6] cf. definition the semantics of [2].

[7] In Reiter's formalization a default is seminormal if it is of the form $\frac{p(d):j(d)\wedge c(d)}{c(d)}$. Since we are only considering ground versions of the defaults the definitions are equivalent.

- $E \subseteq \Gamma'_{\Delta^s}(E)$

The need for this last condition is related to the fact that the operator Ω is the composition of two anti-monotonic operators, that in some cases are equivalent[8]. Thus, in those cases, for every fixpoint E of Ω_Δ there is one other fixpoint $E' = \Gamma'_{\Delta^s}(E)$. The last condition chooses the least of E and E'.

Example 7 For the default theory $\Delta = (\{\frac{:\neg a}{a}\}, \{\})$ there are two fixpoints of Ω_Δ : $E = \{\}$ and $E' = \Gamma'_{\Delta^s}(E) = \{a\}$. Only E obeys the last condition.

Definition 4.9 (Ω–Default semantics) *Let Δ be a default theory, $E = \Omega_\Delta(E)$ an extension, and L a literal.*

- *L is* true *w.r.t. extension E iff $L \in \Omega_\Delta(E)$*
- *L is* false *w.r.t. extension E iff $L \notin \Gamma'_{\Delta^s}(E)$*
- *Otherwise L is* unknown *(or* undefined*)*

Example 8 Consider the default theory $\Delta = (\{\frac{:\neg c}{c}, \frac{:\neg b}{a}, \frac{:\neg a}{b}, \frac{:}{\neg a}\}, \{\})$. Its only extension is $\{\neg a, b\}$. In fact $\Gamma'_{\Delta^s}(\{\neg a, b\}) = \{c, b, \neg a\}$ and $\Gamma_\Delta(E)(\{c, b, \neg a\}) = \{\neg a, b\}$. Thus: $\neg a$ and b are true, c is undefined and a and $\neg b$ are false.

It is easy to see that some theories may have no Ω–extension.

Definition 4.10 (Contradictory theory) *A default theory Δ is contradictory iff it has no Ω–extension.*

Example 9 The theory $\Delta = (\{\frac{:}{a}, \frac{:}{\neg a}\}, \{\})$ has no Ω–extension.

Theorem 4.1 (Ω is monotonic) *If Δ is a noncontradictory theory then Ω_Δ is monotonic.*

Proof: We begin by stating a lemma:

Lemma 4.2 *Let $\Delta = (D, W)$ be a noncontradictory default theory, and $\Delta' = (D \cup \{\frac{:}{L} \mid L \in W\}, \{\})$. E is an Ω–extension of Δ iff is an Ω–extension of Δ'.*

Proof: It is easy to see that every Ω–extension of Δ and of Δ' contains W. Thus for each Ω–extension of one of the theories the set of rules in D applied is the same in the other theory. $\diamondsuit$

Now we prove that if Δ is a noncontradictory default theory then Γ'_Δ is anti-monotonic.

Without loss of generality (cf. lemma 4.2 above) we consider $\Delta = (D, \{\})$. Let A and B be sets of literals such that $A \subseteq B$.

We want to prove $A \subseteq B \Rightarrow \Gamma'_\Delta(B) \subseteq \Gamma'_\Delta(A)$, i.e.:

$$A \subseteq B \Rightarrow \forall d(c(d) \in \Gamma'_\Delta(B) \Rightarrow c(d) \in \Gamma'_\Delta(A)).$$

[8]Example of such cases are theories that are already seminormal, theories without negative conclusion, etc.

Let $c(d)$ be a conclusion of a default rule, such that $c(d) \in \Gamma'_\Delta(B)$. We prove that $c(d) \in \Gamma'_\Delta(A)$ given that $A \subseteq B$.

$$\begin{array}{rcl} c(d) \in \Gamma'_\Delta(B) & \Longrightarrow & \exists_\lambda c(d) \in r^{B,D}_{\lambda+1}(\{\}) \\ & \Longrightarrow & \exists_{\lambda_b \leq \lambda} p(d) \in r^{B,D}_{\lambda_b}(\{\}) \wedge \neg j(d) \cap B = \{\} \end{array}$$

We must prove now:

$$\exists_{\lambda_b} p(d) \in r^{B,D}_{\lambda_b}(\{\}) \wedge \neg j(d) \cap B \Rightarrow \exists_{\lambda_a} p(d) \in r^{A,D}_{\lambda_a}(\{\}) \wedge \neg j(d) \cap A$$

Since $\neg j(d) \cap B = \{\} \Rightarrow \neg j(d) \cap A = \{\}$, it remains to prove that:

$$\exists_{\lambda_b} p(d) \in r^{B,D}_{\lambda_b}(\{\}) \Rightarrow \exists_{\lambda_a} p(d) \in r^{A,D}_{\lambda_a}(\{\})$$

$$\begin{array}{lll} \text{if } p(d) = \{\} & \text{then} & \lambda_a = \lambda_b = 0 \text{ and } c(d) \in r^{B,D}_1(\{\}) \subseteq \Gamma'_\Delta(A) \\ \text{if } p(d) \neq \{\} & \text{then} & \exists_{\lambda' < \lambda_b} p(d) \in r^{B,D}_{\lambda'+1}(\{\}) \end{array}$$

For the last case, and given that λ strictly decreases, the same proof applies recursively[9].

Since Ω_Δ is the composition of two antimonotonic operators, it is monotonic. $\diamondsuit$

Definition 4.11 (Iterative contruction of the least Ω–extension) *In order to obtain a constructive definition for the least (in the set inclusion order sense) Ω–extension of a theory we define the following transfinite sequence $\{E_\alpha\}$:*

$$\begin{array}{rcll} E_0 & = & \{\} & \\ E_{\alpha+1} & = & \Omega(E_\alpha) & \\ E_\delta & = & \bigcup\{E_\alpha \mid \alpha < \delta\} & \textit{for a limit ordinal } \delta \end{array}$$

By theorem 4.1 and according to the properties of monotonic operators, there must exist a smallest λ for the sequence above, such that E_λ is the smallest fixpoint of Ω. If E_λ is a Ω–extension then it is the smallest one. Otherwise, by the proposition below, there are no Ω–extensions for the theory.

Proposition 4.1 *If the least fixpoint E of Ω_Δ is not a Ω–extension of Δ then Δ has no Ω–extensions.*

Proof: Two cases are conceivable for E not to be a Ω–extension: either E has a pair of literals A and $\neg A$, or $\Gamma'_{\Delta^s}(E) \subset E$.

In the first case every fixpoint of Ω_Δ also has that same pair of literals, thus no extension exists for Δ. The second case cannot occur because Γ'_{Δ^s} is antimonotonic. $\diamondsuit$

[9] Note that the expression is similar to the initial one, but with $\lambda' < \lambda_b \leq \lambda$.

Example 10 Consider the default theory Δ of example 8. In order to obtain the least (and only) extension of Δ we build the sequence:

$$\begin{array}{lllllll} E_0 & = & \{\} & & & & \\ E_1 & = & \Gamma'_\Delta(\Gamma'_{\Delta^s}(\{\})) & = & \Gamma'_\Delta(\{c,a,b,\neg a\}) & = & \{\neg a\} \\ E_2 & = & \Gamma'_\Delta(\Gamma'_{\Delta^s}(\{\neg a\})) & = & \Gamma'_\Delta(\{c,b,\neg a\}) & = & \{\neg a, b\} \\ E_3 & = & \Gamma'_\Delta(\Gamma'_{\Delta^s}(\{\neg a, b\})) & = & \Gamma'_\Delta(\{c,b,\neg a\}) & = & \{\neg a, b\} \quad = \quad E_2 \end{array}$$

As E_2 does not contain any pair of complementary literals, and $E_2 \subseteq \Gamma'_{\Delta^s}(E_2)$, it is the least Ω–extension of Δ.

Example 11 Let $\Delta = (\{\frac{:}{a}, \frac{:}{\neg a}\}, \{\})$. Let us build the sequence:

$$\begin{array}{lllllll} E_0 & = & \{\} & & & & \\ E_1 & = & \Gamma'_\Delta(\Gamma'_{\Delta^s}(\{\})) & = & \Gamma'_\Delta(\{a,\neg a\}) & = & \{a, \neg a\} \\ E_2 & = & \Gamma'_\Delta(\Gamma'_{\Delta^s}(\{a, \neg a\})) & = & \Gamma'_\Delta(\{\}) & = & \{a, \neg a\} \quad = \quad E_1 \end{array}$$

As E_1 has a pair of complementary literals Δ has no Ω–extensions.

This new default semantics satisfies all the principles required above (section 4.1).

Theorem 4.3 (Uniqueness of minimal extension) *If Δ has an extension then there is one least extension E_m.*

Proof: The set 2^{AT} is a complete lattice under set inclusion. Since $\Omega_\Delta(L)$ is defined from $2^{AT} \rightarrow 2^{AT}$ and is monotonic then $lfp(\Omega_\Delta)$ always exists. $\diamondsuit$

Theorem 4.4 (Enforcedness) *If E is a Ω–extension then E is enforced.*

Proof: We want to prove that for any default rule $\{p(d), j(d)\} \subseteq E \Rightarrow c(d) \in E$.

Given that E is a Ω–extension, by definition $\Omega(E) \subseteq \Gamma'_{\Delta^s}(E)$ holds. Thus for any default rule:

$$\{p(d), j(d)\} \subseteq E \Rightarrow \{p(d), j(d)\} \subseteq \Gamma'_{\Delta^s}(E), \text{ and } \neg j(d) \notin E.$$

By definition $p(d) \in \Gamma'_{\Delta^s}(E) \Leftrightarrow \exists_\lambda p(d) \in r_\lambda^{\Gamma'_{\Delta^s}(E),D}(\{\})$.

Since $\neg j(d) \notin E$, it follows that: $c(d) \in r_{\lambda+1}^{\Gamma'_{\Delta^s}(E),D}(\{\})$. Thus:

$$c(d) \in r_\infty^{\Gamma'_{\Delta^s}(E),D}(\{\}) = \Gamma'_\Delta(\Gamma'_{\Delta^s}(E)) = E$$

$\diamondsuit$

Corollary 1 *If E is an Ω–extension of Δ then for any $d = \frac{:}{c(d)} \in \Delta$, $c(d) \in E$.*

Proof: Follows directly from enforcedness for true prerequisites and justifications. $\diamondsuit$

Theorem 4.5 (Modularity) *Let $L(\Delta_1)$ and $L(\Delta_2)$ be the languages of two default theories. If $L(\Delta_1) \cap L(\Delta_2) = \{\}$ then, for any corresponding extensions E_1 and E_2, there always exists an extension E of $\Delta = \Delta_1 \cup \Delta_2$ such that $E = E_1 \cup E_2$.*

Proof: Since the languages are disjoint, the rules of Δ_1 and Δ_2 do not interact on that count. Additionally, since the Cn operator is not applied, we never obtain the whole set of literals as a result of Γ'_{Δ^s}, and hence they do not interact on that count either. $\diamondsuit$

4.3 Comparison with Reiter's semantics

Comparing ous with Reiter's semantics for default theories, we prove that for theories whose language is such that our semantics can be applied the former is a generalization of the latter, in the sense that whenever Reiter's semantics (Γ–extension) gives a meaning to a theory (i.e. the theory has at least one Γ–extension), our semantics also provides one.

Moreover, when both semantics give meaning to a theory our semantics is sound w.r.t. the intersection of all Γ–extensions. Thus we provide a monotonic fixpoint operator for computing a subset of the intersection of all Γ–extensions. With that purpose we begin by stating a theorem:

Theorem 4.6 *Consider a theory Δ such that our semantics is defined. Then every Γ–extension is a Ω–extension.*

Proof: We begin by stating two lemmas

Lemma 4.7 $E = \Gamma_\Delta(E) \Rightarrow E = \Gamma'_{\Delta^s}(E)$.

Proof: By definition of Γ_Δ, $E = \Gamma_\Delta(E) \Leftrightarrow (\forall_{d \in D}\ p(d) \in E \wedge \neg j(d) \cap E = 0$.
But:

Since	$(\forall_{d \in D}\ p(d) \in E \wedge \neg j(d) \cap E = 0$	$\Rightarrow c(d) \in E)$
then	$(\forall_{d \in D}\ p(d) \in E \wedge \neg j(d) \cap E = 0 \wedge \neg c(d) \cap E = 0$	$\Rightarrow c(d) \in E)$

i.e., by definition, $E = \Gamma'_{\Delta^s}(E)$. $\diamondsuit$

Lemma 4.8 $E = \Gamma_\Delta(E) \Rightarrow E = \Gamma'_\Delta(E)$.

Proof: Similar to the proof of lemma 4.7. $\diamondsuit$

Now we prove that for an E such that $E = \Gamma_\Delta(E)$, $E = \Omega_\Delta(E)$ holds.

By definition, $\Omega_\Delta(E) = \Gamma'_\Delta(\Gamma'_{\Delta^s}(E))$. By lemma 4.7, $\Omega_\Delta(E) = \Gamma'_\Delta(E)$, and by lemma 4.8, $\Gamma'_\Delta(E) = E$.

For E to be a Ω–extension two more conditions must hold: E cannot have a pair of complementary literals, which holds because it is a Γ–extension; and $E \subseteq \Gamma'_{\Delta^s}(E)$. It is easy to see that the latter condition holds given that $E = \Gamma_\Delta(E)$. $\diamondsuit$

Theorem 4.9 (Generalization of Reiter's semantics) *If a theory Δ has at least one Γ–extension, it has at least one Ω–extension.*

Theorem 4.10 (Soundness wrt to Reiter's semantics) *If a theory Δ has a Γ–extension, whenever L belongs to the least Ω–extension it also belongs to the intersection of all Γ–extensions.*

Proof: Follows directly from the theorem and the property that the least Ω–extension is the intersection of all Ω–extensions. $\diamondsuit$

4.4 Comparison with Przymusinski's semantics

We now draw some comparisons with stationary extensions [13]. It is not the case that every stationary extension is a Ω–extension since, as noted above, nonmodular or nonenforced stationary extensions are not Ω–extensions. As shown in the example below, it is also not the case that every Ω–extension is a stationary extension.

Example 12 Let $\Delta = (\{\frac{:\neg b}{c}, \frac{:\neg a}{b}, \frac{:\neg a}{a}, \frac{:}{\neg b}\}, \{\})$. The only Ω–extension of Δ is $\{c, \neg b\}$. Note that this is not a stationary extension.

However, in a large class of cases these semantics coincide. In particular:

Proposition 4.2 *If for every default $d = \frac{p(d):j(d)}{c(d)}$ $c(d)$ is a positive literal then Ω coincides with Γ^2_Δ.*

5 Relation between the Semantics of Default Theories and Logic Programs with Explicit Negation

In this section we prove the exact correspondence among the Ω–extensions and the XSMs of extended logic programs.

Definition 5.1 (Program corresponding to a default theory) *Let $\Delta = (D, \{\})$ be a default theory. We say that an extended logic program P corresponds to Δ iff:*

- *For every default of the form $\frac{\{a_1,\ldots,a_n\}\ :\ \{b_1,\ldots,b_m\}}{C} \in \Delta$ there exists a rule $c \leftarrow a_1, \ldots, a_n, \sim\neg b_1, \ldots, \sim\neg b_m \in P$, where b_j denotes the complement of $\neg b_j$.*
- *No rules other than these belong to P.*

Definition 5.2 (Interpretation corresponding to a default context) *An interpretation I of a program P corresponds to a context E of the corresponding default theory T iff for every classical literal L of P (and literal L of T):*

- *$I(L) = 1$ iff $L \in E$ and $L \in \Gamma'_{\Delta^s}(E)$.*
- *$I(L) = \frac{1}{2}$ iff $L \notin E$ and $L \in \Gamma'_{\Delta^s}(E)$.*
- *$I(L) = 0$ iff $L \notin E$ and $L \notin \Gamma'_{\Delta^s}(E)$.*

Theorem 5.1 (Correspondence) *Let $\Delta = (D, \{\})$ be a default theory corresponding to the program P. E is a Ω-extension of Δ iff the interpretation I corresponding to E is a XSM of P. (Proof ahead)*

According to this theorem we can say that explicit negation is nothing but classical negation in default theories. As Ω default semantics is a generalization of Γ default semantics (in the sense of theorems 4.9 and 4.10), and since answer-sets semantics correspond to Γ default semantics [6], it turns out that answer-sets semantics (and hence the semantics defined in [17]) are special cases of $WFSX$ semantics in the same sense.

Example 13 Consider program $P = \{c \leftarrow \sim c, \quad a \leftarrow \sim b, \quad b \leftarrow \sim a, \quad \neg a \leftarrow\}$. The corresponding default theory is $\Delta = (\{\frac{:\neg c}{c}, \frac{:\neg b}{a}, \frac{:\neg a}{b}, \frac{:}{\neg a}\}, \{\})$.

As calculated in example 8, the only Ω–extension of Δ is $E = \{\neg a, b\}$ and $\Gamma'_{\Delta^s}(E) = \{\neg a, b, c\}$. The XSM corresponding to this extension is:

$$M = \{\neg a, \sim a, b, \sim\neg b, \sim\neg c\}^{10}.$$

It is easy to verify that M is the only XSM of P.

Proof of Theorem 5.1: We begin by stating some propositions useful in the sequel.

Proposition 5.1 *Let $\Delta = (D, \{\})$ be a default theory and E a context such that $\Gamma'_\Delta(E)$ is noncontradictory. Then:*

$$L \in \Gamma'_\Delta(E) \Leftrightarrow \exists \frac{\{b_1, \ldots, b_n\} : \{c_1, \ldots, c_m\}}{L} \in D, \ \forall i, j \ b_i \in \Gamma'_\Delta(E) \wedge \neg c_j \notin E$$

Proof: It is easy to see that under these conditions $\Gamma'_\Delta(E) = \Gamma_\Delta(E)$. Thus the proof follows from properties of the Γ_Δ operator. $\diamondsuit$

Proposition 5.2 *Let E be an extension of a default theory $\Delta = (D, \{\})$. Then:*

$$L \in \Omega(E) \Leftrightarrow \exists \frac{\{b_1, \ldots, b_n\} : \{\neg c_1, \ldots, \neg c_m\}}{L} \in D \text{ such that}$$

$$\forall i, j, \ b_i \in E \wedge b_i \in \Gamma'_{\Delta^s}(E) \wedge c_j \notin \Gamma'_{\Delta^s}(E).$$

Proof: By definition of Γ'_Δ, and given that $W = \{\}$, it follows from proposition 5.1 that for $L \in \Omega(E)$ there must exist at least one default in D applied in the second step, i.e. with all prerequesites in $\Omega(E)$ and all negations of justifications not in $\Gamma'_{\Delta^s}(E)$. By hypothesis E is an extension; thus $E = \Omega(E)$ and $E \subseteq \Gamma'_{\Delta^s}(E)$; so for such a rule all prerequesites are in E and in $\Gamma'_{\Delta^s}(E)$, and all negations of justifications are not in $\Gamma'_{\Delta^s}(E)$. $\diamondsuit$

Proposition 5.3 *Let E be an extension of a default theory $\Delta = (D, \{\})$. Then:*

$$L \notin E \Rightarrow \forall \frac{\{b_1, \ldots, b_n\} : \{\neg c_1, \ldots, \neg c_m\}}{L} \in D, \ \exists i, j, \ b_i \notin E \vee c_j \in \Gamma'_{\Delta^s}(E)$$

[10] Note that c is undefined in M.

Proof: If $L \notin E$ then, given that E is an extension, $L \notin \Omega_\Delta(E)$. Thus no default rule for L is applicable in the second step, i.e. given that $W = \{\}$, and by proposition 5.1, no rule with conclusion L is such that all its prerequisites are in $\Omega_\Delta(E)$ and no negation of a justification is in $\Gamma'_{\Delta^s}(E)$. $\diamondsuit$

Proposition 5.4 *Let E be an extension of a default theory $\Delta = (D, \{\})$. Then:*

$$L \notin \Gamma'_{\Delta^s}(E) \Leftrightarrow \forall \frac{\{b_1,\ldots,b_n\} : \{\neg c_1,\ldots,\neg c_m\}}{L} \in D,$$

$$\exists i, j,\ b_i \notin \Gamma'_{\Delta^s}(E) \vee c_j \in E \vee \neg L \in E$$

Proof: Similar to the proof of 5.3 but now applied to the first step, which imposes the use of seminormal defaults. Thus the need for $\neg L \in E$. $\diamondsuit$

We now prove the main theorem:

($\rightarrow$) E is a Ω-extension of $\Delta \Rightarrow I$ is a XSM of P.

Here we must prove that for any (classical and default) literal F, $F \in I \Leftrightarrow F \in \Phi(I)$. We do this in three parts: for any classical literal L, $L \in I \Rightarrow L \in \Phi(I)$; $L \notin I \Rightarrow L \notin \Phi(I)$; $\sim L \in I \Leftrightarrow \sim L \in \Phi(I)$.

Each of these proofs proceeds by: translating conditions in I into conditions in E via correspondence; finding conditions in Δ given the conditions in E, and the fact that E is an extension; translating conditions in Δ into conditions in P via correspondence; using those conditions in P to determine the result of operator Φ.

$L \in I \Leftrightarrow I(L) = 1 \Rightarrow L \in E \Leftrightarrow L \in \Omega_\Delta(E)$, because I corresponds to E and E is a Ω-extension.

By proposition 5.2 $L \in \Omega_\Delta(E) \Leftrightarrow \exists \frac{\{b_1,\ldots,b_n\} : \{\neg c_1,\ldots,\neg c_m\}}{L} \in D$ such that $\forall i, b_i \in E \wedge b_i \in \Gamma'_{\Delta^s}(E)$ and $\forall j, c_j \notin \Gamma'_{\Delta^s}(E)$.

By translating, via the correspondence definitions, the default and the conditions on E into a rule and conditions on I:

$L \in E \Rightarrow \exists L \leftarrow b_1, \ldots, b_n, \sim c_1, \ldots, c_m \in P$ such that $\forall i, I(b_i) = 1$ and $\forall j, I(c_j) = 0 \Rightarrow L \in least(\frac{P}{I})$, by properties of $least(\frac{P}{I})$.

Given that the operator Coh does not delete literals from I, $L \in I \Rightarrow L \in \Phi(I)$.

$L \notin I \Leftrightarrow L \notin E$, because I corresponds to E.

$L \notin E \Rightarrow \forall \frac{\{b_1,\ldots,b_n\} : \{\neg c_1,\ldots,\neg c_m\}}{L} \in D$ then either a $b_i \notin E$ or a $c_j \in \Gamma'_{\Delta^s}(E)$, by proposition 5.3.

Translating, via the correspondence definitions, the default and the conditions on E into a rule and conditions on I:

$L \notin E \Rightarrow \forall L \leftarrow b_1, \ldots, b_n, \sim c_1, \ldots, c_m \in P$ then $\exists i, j\ I(b_i) \neq 1 \vee I(c_j) \neq 0 \Rightarrow L \notin least(\frac{P}{I})$, by properties of $least(\frac{P}{I})$.

Given that the operator Coh does not add classical literals to I, $L \notin I \Rightarrow L \notin \Phi(I)$.

$\sim L \in I \Leftrightarrow L \notin \Gamma'_{\Delta^s}(E)$, since E corresponds to I.

$L \notin \Gamma'_{\Delta^s}(E) \Leftrightarrow \forall \frac{\{b_1,\ldots,b_n\} : \{\neg c_1,\ldots,\neg c_m\}}{L} \in D$ then $\exists i, j\ b_i \notin \Gamma'_{\Delta^s}(E) \vee c_j \in E \vee \neg L \in E$, by proposition 5.4.

$L \notin \Gamma'_{\Delta^s}(E) \Leftrightarrow (\forall L \leftarrow b_1,\ldots,b_n,\sim c_1,\ldots,c_m \in P$ then $\exists i,j\ I(b_i)=0 \vee I(c_j)=1) \vee \neg L \in E$.

$\sim L \in I \Leftrightarrow \sim L \in least(\frac{P}{I}) \vee \neg L \in E$ $(*)$, by properties of the least operator.

It was proven before that $\neg L \in E \Leftrightarrow \exists \neg L \leftarrow b_1,\ldots,b_n,\sim c_1,\ldots,\sim c_m \in P$ then $\exists i,j\ I(b_i)=1 \vee I(c_j)=0)$.

By properties of $least(\frac{P}{I})$, $\neg L \in E \Leftrightarrow \neg L \in least(\frac{P}{I})$.

Using correspondence, we can simplify the equivalence $(*)$ to: $\sim L \in I \Leftrightarrow \sim L \in least(\frac{P}{I}) \vee \neg L \in least(\frac{P}{I}) \Leftrightarrow \sim L \in \Phi(I)$, this last equivalence being due to the definitions of operators Coh and Φ.

$(\leftarrow)$ I is a XSM of $P \Rightarrow E$ is a Ω-extension of T.

By definition of correspondence between interpretations and contexts, it is easy to see that E is consistent and $E \subseteq \Gamma'_{\Delta^s}(E)$. So we only have to prove that $E = \Omega_\Delta(E)$. We do this by proving that $\forall L\ L \in E \Leftrightarrow L \in \Omega_\Delta(E)$.

$L \in E \Leftrightarrow I(L)=1$ by definition of corresponding context.

$I(L)=1 \Leftrightarrow \exists L \leftarrow b_1,\ldots,b_n,\sim c_1,\ldots,\sim c_m \in P$, where $n,m \geq 0$ such that $\forall i\ I(b_i)=1$ and $\forall j\ I(c_j)=0$, because I is a XSM of P.

By translating, via the correspondence definitions, the rule and the conditions on I into a default and conditions on E:

$I(L)=1 \Leftrightarrow \exists \frac{\{b_1,\ldots,b_n\}\ :\ \{\neg c_1,\ldots,\neg c_m\}}{L} \in D$ such that $\forall i\ b_i \in E \wedge b_i \in \Gamma'_{\Delta^s}(E)$ and $\forall j\ c_j \notin E \wedge c_j \notin \Gamma'_{\Delta^s}(E)$

Given that such a rule exists under such conditions, it follows easily from theorem 5.1 that $L \in E \Leftrightarrow L \in \Omega_\Delta(E)$. $\diamondsuit$

Acknowledgements

We thank ESPRIT BRA COMPULOG (no. 3012), Instituto Nacional de Investigação Científica, Junta Nacional de Investigação Científica e Tecnológica, and Gabinete de Filosofia do Conhecimento for their support.

References

[1] C. Baral and V. S. Subrahmanian. Stable and extension class theory for logic programs and default logics. In *International Workshop on Nonmonotonic Reasoning*, 1990.

[2] C. Baral and V. S. Subrahmanian. Dualities between alternative semantics for logic programming and nonmonotonic reasoning. In A. Nerode, W. Marek, and V. S. Subrahmanian, editors, *Logic Programming and NonMonotonic Reasoning'91*. MIT Press, 1991.

[3] N. Bidoit and C. Froidevaux. Minimalism subsumes default logic and circumscription in stratified logic programming. In *Symposium on Principles of Database Systems*. ACM SIGACT-SIGMOD, 1987.

[4] N. Bidoit and C. Froidevaux. General logic databases and programs: default logic semantics and stratification. *Journal of Information and Computation*, 1988.

[5] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In R. A. Kowalski and K. A. Bowen, editors, *5th International Conference on Logic Programming*, pages 1070–1080. MIT Press, 1988.

[6] M. Gelfond and V. Lifschitz. Logic programs with classical negation. In Warren and Szeredi, editors, *7th International Conference on Logic Programming*, pages 579–597. MIT Press, 1990.

[7] A. Marek and M. Truszczynski. Stable semantics for logic programs and default theories. In *North American Conference on Logic Programming'89*. MIT Press, 1989.

[8] L. M. Pereira and J. J. Alferes. Well founded semantics for logic programs with explicit negation. In *European Conference on Artificial Inteligence'92*. John Wiley & Sons, Ltd, 1992. *To appear.*

[9] L. M. Pereira, J. N. Aparício, and J. J. Alferes. Counterfactual reasoning based on revising assumptions. In Ueda and Saraswat, editors, *International Logic Programming Symposium'91*. MIT Press, 1991.

[10] L. M. Pereira, J. N. Aparício, and J. J. Alferes. Hypothetical reasoning with well founded semantics. In B. Mayoh, editor, *Scandinavian Conference on AI'91*. IOS Press, 1991.

[11] L. M. Pereira, J. N. Aparício, and J. J. Alferes. Nonmonotonic reasoning with well founded semantics. In Koichi Furukawa, editor, *8th International Conference on Logic Programming'91*, pages 475–489. MIT Press, 1991.

[12] H. Przymusinska and T. Przymusinski. *Semantic Issues in Deductive Databases and Logic Programs.* Formal Techniques in Artificial Intelligence. North Holland, 1990.

[13] H. Przymusinska and T. Przymusinski. Stationary default extensions. Technical report, California Polytechnic at Pomona and University of California at Riverside, 1991.

[14] T. Przymusinski. Extended stable semantics for normal and disjunctive programs. In Waren and Szeredi, editors, *7th International Conference on Logic Programming*, pages 459–477. MIT Press, 1990.

[15] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:68–93, 1980.

[16] A. Van Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *Journal of ACM*, pages 221–230, 1990.

[17] G. Wagner. A database needs two kinds of negation. In B. Thalheim, J. Demetrovics, and H-D. Gerhardt, editors, *MFDBS'91*, pages 357–371. Springer-Verlag, 1991.

A Default Theory Review

Here we review some known default theory semantics. We begin by reviewing Reiter's classical default logic [15]. We then review (partly following [2]) the well-founded [2] and stationary [13] default logics, which correspond respectively to the well founded and stationary semantics of (nonextended) logic programs.

A.1 Classical Default Theories

Definition A.1 **($R^{E,D}(S)$)** *Suppose (D,W) is a default theory, S is a set of formulas, and E is a set of formulas called the* context. *The operator $R^{E,D}(S)$ mapping formulas into formulas is defined as:*

$$R^{E,D}(S) = Cn(S \cup c(d) | d \in D, p(d) \in S, \neg j(d) \cap E = \{\}) \tag{2}$$

where Cn denotes the Tarskian consequence operator, and $\neg j(d)$ denotes the set $\{\neg\beta | \beta \in j(d)\}$.

Note that the operator above enables to add the consequences of those defaults whose justifications are consistent with context E.

Definition A.2 **($\Gamma_\Delta(E)$)** *Suppose (D,W) is a default theory and E is a context. Then*

$$\begin{array}{l} R_0^{E,D}(W) = Cn(W) \\ R_{n+1}^{E,D}(W) = R^{E,D}(R_n^{E,D}(W)) \\ R_\infty^{E,D}(W) = \bigcup_{n=0}^{\infty} R_n^{E,D}(W) \end{array}$$

As noted in [7], if (D,W) is a default theory $R_\infty^{E,D}(W)$ is identical to Reiter's operator $\Gamma_\Delta(E)$ [15].

Definition A.3 (Default extension) *E is a classical extension of a default theory (D,W) iff $E = R_\infty^{E,D}(W)$ or, equivalently, iff $E = \Gamma_\Delta(E)$.*

One problem of classical default theory is that it may have multiple extensions and then no single minimal extension gives meaning to the theory, or no extensions at all (and no meaning is given), in cases where a definite meaning is expected.

Example 14 The default theory $W = \{p\}, D = \{\frac{: \neg q}{q}\}$ has no extensions. However p is a fact, and we would expect it to be true.

A.2 Well Founded and Stationary Default Semantics of General Logic Programs

We review here two approaches which relate general logic programs with default theories.

Baral et. al. [2] introduce the well founded semantics for default theories giving a meaning to default theories with multiple extensions. Furthermore, the semantics is defined for all theories, identifying a single extension for each theory.

Let $\Delta = (D,W)$ be a default theory, and let $\Gamma_\Delta(E)$ be as above. Since $\Gamma_\Delta(E)$ is antimonotonic $\Gamma_\Delta^2(E)$ is monotonic [2].

Definition A.4 (Well founded semantics) *[2]*[11]

- *A formula F is* true *in a default theory Δ with respect to the well–founded semantics iff $F \in lfp(\Gamma^2)$.*
- *F is* false *in Δ w.r.t. the well founded semantics iff $F \notin gfp(\Gamma^2)$.*
- *Otherwise F is said to be* unknown *(or* undefined*).*

This semantics is defined for all theories and is equivalent to the Well Founded Model semantics [16] of general logic programs.

Definition A.5 (Stationary extension) *[13]*
Given a default theory Δ, E is a stationary default extension iff:

- $E = \Gamma^2_\Delta(E)$
- $E \subseteq \Gamma_\Delta(E)$

Definition A.6 (Stationary default semantics) *[13]*

- *A formula L is true in E iff $L \in \Gamma^2_\Delta(E) = E$.*
- *A formula L is false in E iff $L \notin \Gamma_\Delta(E)$.*
- *Otherwise L is said to be* unknown *(or* undefined*).*

Remark A.1 *Note that every default theory has at least one stationary default extension. The least stationary default extension always exists, and corresponds to the well founded semantics for default theories above. Indeed, $\Gamma^2_\Delta = lfp(\Gamma^2_\Delta)$, and $\Gamma_\Delta = gfp(\Gamma^2_\Delta)$, whenever $E \subseteq \Gamma_\Delta(E)$(i.e. $lfp(\Gamma^2_\Delta) \subseteq gfp(\Gamma^2_\Delta)$). Moreover, the least stationary default extension can be computed by iterating the monotonic operator Γ^2_Δ.*

Example 15 Consider the default theory of example 14. We have $\Gamma_\Delta(\{p\}) = \{p, q\}$ and $\Gamma^2_\Delta(\{p\}) = \{p\}$. p is true in the theory Δ.

[11] In [2] the notation used is slightly different but the definitions are equivalent.