Belief, Provability, and Logic Programs

José Júlio Alferes
D. Matemática, U. Évora, and
CRIA Uninova
2825 Monte da Caparica
Portugal
(jja@fct.unl.pt)

Luís Moniz Pereira
DCS, U. Nova de Lisboa, and
CRIA Uninova
2825 Monte da Caparica
Portugal
(lmp@fct.unl.pt)

ABSTRACT. *The main goal of this paper is to establish a nonmonotonic epistemic logic $\mathcal{EB}$ with two modalities – provability and belief – capable of expressing and comparing a variety of known semantics for extended logic programs, and clarify their meaning. In particular we present here, for the first time, embeddings into epistemic logic of logic programs extended with a second kind of negation under the well–founded semantics, and contrast them to the recent embeddings into autoepistemic logics of such programs under stable models based semantics.*

Because of the newly established relationship between our epistemic logic $\mathcal{EB}$ and extended program semantics, the former benefits from the procedures and implementations of the latter, and can be applied to at least the same class of AI problems that the latter can. Moreover, one issue of epistemic logic introduced here, belief revision, can profit from adapting techniques employed by the latter for contradiction removal.

Furthermore, the language of the epistemic logic presented here being more general than that of extended programs, it offers a basic tool for further generalizations of the latter, for instance regarding disjunction and modal operators.

Introduction

The relationships between logic programming and several nonmonotonic reasoning formalisms bring them mutual benefits. Nonmonotonic formalisms provide semantics for logic programs, and help understand how these can express and compute solutions to AI problems. Conversely, the nonmonotonic formalisms benefit from the procedures and implementations of logic programming. Also, relations among nonmonotonic formalisms have been studied via logic programming shunting.

For normal logic programs the bridge to default theories [Rei80] was first made in [BF87]. In [EK89] negation as failure of normal programs was first formalized as abduction, and in [Dun91] was extended to capture both stable models [GL88] and the well-founded semantics (WFS) of normal logic programs [GRS91].

The view of logic programs as autoepistemic theories [Moo85] first appeared in [Gel87], which envisages every literal $not\ L$ of logic programs as $\sim\mathcal{L}L$, i.e. $not\ L$

has the epistemic reading: “there is no reason to believe in L”[1]. In [Bon92] a variety of translations of negation as failure by belief literals are studied, in order to show how different logic programming semantics can be obtained from autoepistemic logics (AELs). In [Prz93], Przymusinski assigns to $not\ L$ the translation $\mathcal{B} \sim L$, with the reading “L is believed to be false”.

Several authors have stressed the importance of extending logic programming with a second kind of negation $\neg$, in addition to default negation, for use in deductive databases, knowledge representation, and nonmonotonic reasoning [GL90, KS90, PAA91b, Wag91]. Different semantics for extended logic programs with $\neg$-negation have appeared [DR91, GL90, KS90, PA92, Prz90, Prz91b, Wag91]. [AP92] contrasts some of these, where distinct meanings of $\neg$-negation are identified: classical, strong and explicit. It is also argued that explicit negation is preferable.

Some work exists comparing extended logic programs semantics and nonmonotonic reasoning formalisms. In [GL90] the answer-sets semantics for extended programs is introduced and compared to default theories. A comparison between the WFS with explicit negation (*WFSX*) [PA92] and default theories is given in [PAA92]. *WFSX* is captured within an abductive framework in [ADP93].

As noted by [Che93, LS93b, MT93], Gelfond’s translation cannot be generalized to extended programs.

Example 1 According to Gelfond’s translation, P :

$$\begin{array}{rcl} a & \leftarrow & b \\ \neg a & & \end{array}$$

is rendered as the theory

$$T = \{b \Rightarrow a;\ \sim a\}.$$

This theory entails $\{\sim a, \sim b\}$, but the semantics of P under most of the approaches (e.g. under *WFSX* and answer-sets) is $\{\neg a\}$.

A suitable translation between extended programs with answer-sets semantics and reflexive AEL theories was proposed independently in [LS93b] and [MT93]. Reflexive AEL, introduced in [Sch91], views the operator $\mathcal{L}$ as “is known” instead of the “is believed” of Moore’s AEL [Moo85][2]. The translation renders an objective literal A (resp. $\neg A$) as $\mathcal{L}A$ (resp. $\mathcal{L} \sim A$, where $\sim$ denotes classical negation), i.e. “A is known to be true” (resp. “A is known to be false”), and renders $not\ L$ as $\mathcal{L} \sim \mathcal{L}L$, i.e. “it is known that L is not known”. In [LS93b, MT93] the authors prove that the answer-sets of an extended program correspond to the reflexive expansions of its translation. Equivalently, the embedding of extended programs into reflexive AEL can also be defined for (non-reflexive) AEL [LS93b, MT93], by translating any objective literal L into $L \wedge \mathcal{L}L$. This translation was proposed in [Che93] too.

[1] Referred to here as Gelfond’s translation.
[2] Roughly, this is achieved by adding $F \equiv \mathcal{L}F$, instead of just $\mathcal{L}F$, when F holds.

In [Pea93], the author surmises a translation also equivalent to the ones above. This translation is justified by first relating answer-sets to constructive logics with strong negation [Nel49], and then using the already known translation of the latter into nonmonotnic S4.

The embedding of stable models semantics into AEL was generalized to WFS in [Prz91a], using Gelfond's translation, but where Generalized Closed World Assumption (GCWA) [Min87] replaces the Closed World Assumption (CWA) [Rei78] in what regards the adoption of default literals. No study of embeddings of WFS with $\neg$-negation exists to date. One main purpose of this paper is to remedy this. Significantly, the embedding proposed in [Che93, LS93b, MT93, Pea93] does not generalize to extended programs under WFS.

Example 2 The program:

$$P = \{a \leftarrow not\ a\}$$

translates into the non-reflexive AEL theory

$$T = \{\sim\mathcal{L}a \Rightarrow \mathcal{L}a \wedge a\} = \{\mathcal{L}a\}.$$

It is easy to see that this theory has no expansion, even when GCWA is taken up instead of CWA. The same goes for the reflexive AEL translation.

Indeed, that translation is too specific, and can only be applied to stable models based semantics (i.e. that are two-valued).

In contradistinction, our stance is that, for greater generality, the second kind of negation introduced in logic programming represents and requires, for translation into some epistemic logic, an additional modality other than the one necessary for interpreting negation by default[3]. In our view, an objective literal $\neg A$ (resp. A) should be read

"A is proven false"

denoted by $\mathcal{E} \sim A$ (resp. "A is proven true"); and $not\ L$ should be read "it is believed that L is not proven", denoted by $\mathcal{B} \sim \mathcal{E}L$. Thus, $\mathcal{E}$ refers to epistemic knowledge as defined by propositional provability, and relates to the consistency modality $\mathcal{M}$ by $\mathcal{E} \equiv \sim\mathcal{M}\sim$. The belief operator of this logic is $\mathcal{B}$, and is inspired by the one introduced in [Prz93].

The main goal of this paper is to define, in section 1, an AEL augmented with the modality $\mathcal{E}$ which is capable of expressing and comparing various semantics of extended programs. The flexibility and generality of our approach are brought out in section 2, by establishing how different notions of provability and knowledge, and different semantics for extended programs are captured by it, and so providing for a

[3]In [Lif92] the author also proposes a bi-modal logic (MBNF) for interpreting extended logic programs. There is a MBNF rendering of answer-sets which, as shown in [Che93, LS93b], is equivalent to the AEL-unimodal translations, already discussed above, that express answer-sets too.

better understanding of the different kinds of negation. The improved generality of our AEL language provides a tool for examining further generalizations of extended logic programming. This is discussed in section 3. For the sake of self-sufficiency, in appendix we present the definitions of Answer-Sets, and *WFSX* semantics.

1 A logic of belief and provability

In this section we define an epistemic logic, $\mathcal{EB}$, with provability and belief modalities, and show how it captures the *WFSX* semantics for extended logic programming [PA92], which extends WFS with explicit negation [AP92], in addition to default negation.

We begin by considering definite extended logic programs only (i.e. extended programs without negation by default), and by defining a modal logic to interpret such programs. We then extend this logic to deal with belief propositions. Finally, we relate the $\mathcal{EB}$ logic to the full language of *WFSX*.

1.1 Provability in extended definite programs

To motivate and make clear the meaning of the provability modality, we begin with the simpler problem of how to capture the meaning of extended programs without negation by default, i.e. sets of rules of the form:

$$L_0 \leftarrow L_1, \ldots, L_n \quad n \geq 0 \tag{1}$$

where each L_i is an atom A or its explicit negation $\neg A$. Without loss of generality, as in [PP90], we assume that all rules are ground, and programs may be infinite sets of such rules.

The semantics of these programs is desireably monotonic, and must be noncontrapositive, i.e. distinguish between $a \leftarrow b$ and $\neg b \leftarrow \neg a$, so that rules can be viewed as (unidirectional) "inference rules"; Gelfond's translation does not capture this distinction: both rules translate to $b \Rightarrow a$.

Example 3 In example 1, notice how $\sim b$ is derived in T via the contrapositive of the first rule.

The cause of the problem is that $\neg A$ translates into "A is false", and the rule connective $\leftarrow$ into material implication. In contrast, the semantics of extended logic programs wants to interpret $\neg A$ as "A is provenly false", in a grounded sense, and $\leftarrow$ as an inference rule. To capture this meaning we introduce the modal operator $\mathcal{E}$, referring to *(propositional) "provability"*, or *"epistemic knowledge"*, and accordingly translate rule (1) into:

$$\mathcal{E}L_1 \wedge \ldots \wedge \mathcal{E}L_n \Rightarrow \mathcal{E}L_0 \tag{2}$$

where any explicitly negated literal $\neg A$ is translated into $\mathcal{E} \sim A$ and reads "A is provenly false", and any atom A is translated into $\mathcal{E}A$ and reads "A is provenly true".

This translation directly captures the intuitive meaning of a rule

"if all $L_1, \ldots, L_n$ are provable then L_0 is provable" and does not conflate contrapositives: $a \leftarrow b$ becomes $\mathcal{E}b \Rightarrow \mathcal{E}a$, whilst $\neg b \leftarrow \neg a$ is rendered as $\mathcal{E} \sim a \Rightarrow \mathcal{E} \sim b$.

Note the similarities to the translation defined in [LS93b, MT93] into reflexive AEL, where an atom A is translated into $\mathcal{L}A$, and $\neg A$ into $\mathcal{L} \sim A$, where $\mathcal{L}$ is the knowledge operator of modal logic **SW5**.

We need to assume little about $\mathcal{E}$, and this guarantees flexibility. $\mathcal{E}$ is defined as the necessity operator of the smallest normal modal system, modal logic **K**. This logic includes only[4], modus ponens, and:

$$\text{Necessitation: } \frac{F}{\mathcal{E}F}$$

$$\text{Distribution over conjunctions: } \mathcal{E}(F \wedge G) \equiv \mathcal{E}F \wedge \mathcal{E}G$$

$$K: \ \mathcal{E}(F \Rightarrow G) \Rightarrow (\mathcal{E}F \Rightarrow \mathcal{E}G)$$

In logic **K**, $\mathcal{E}$ is the dual of the modal consistency operator $\mathcal{M}$, i.e. $\mathcal{E} \equiv \sim\mathcal{M} \sim$. This weak modal logic, although sufficient for *WFSX* when combined with a belief modality and nonmonotonicity (as shown below), can also express other (stronger) meanings of $\mathcal{E}$ just by introducing more axioms for it. In section 2, in particular, we interpret $\mathcal{E}$ as knowledge by introducing, as usual, the additional axioms for the stronger logic **SW5**.

Since at this stage we are simply interested in the semantics of monotonic (definite) extended programs, we do not require yet a nonmonotonic version of this logic.

Above we said that translation (2) can capture the semantics of extended logic programs. The next theorem makes this statement precise for answer–sets and *WFSX* semantics. It generalizes for almost every semantics of extended logic programs, the only exception being, to our knowledge, the "stationary semantics with classical negation" defined in [Prz91b], which is contrapositive.

Theorem 1.1 *Let P be an extended logic program, and T the theory obtained from P by means of translation (2). If $T \vdash_K \mathcal{E}A \wedge \mathcal{E} \sim A$, for no atom A, then:*

$$\begin{array}{lclcl} T \vdash_K \mathcal{E}A & \equiv & P \models_{AS} A & \equiv & P \models_{\text{WFSX}} A \\ T \vdash_K \mathcal{E} \sim A & \equiv & P \models_{AS} \neg A & \equiv & P \models_{\text{WFSX}} \neg A \end{array}$$

where $\vdash_{\mathcal{S}}$ denotes, as usual, the consequence relation in modal logic $\mathcal{S}$ (in this case K), and $P \models_{AS} L$ (resp. $P \models_{\text{WFSX}} L$) means that L belongs to all answer–sets (resp. all WFSX *partial stable models) of P.*

Otherwise, the only answer–set is the set of all objective literals, and P is contradictory wrt to WFSX.

1.2 Belief and provability

Besides explicit negation, extended logic programs also allow negation by default, which is nonmonotonic and usually understood as a belief proposition. Thus, we need

[4] For a precise definition of logic **K** and its properties see [Che80, HC84].

to enlarge modal logic **K** with a nonmonotonic belief operator.

Before tackling the more general problem, we begin by defining what beliefs follow from definite extended programs. Such programs are readily translatable into sets of Horn clauses, thereby possessing a unique minimal model. So, as a first approach consider:

"the agent believes in a formula if it belongs to the minimal model of the theory"

i.e.

$$\text{if } T \models_{min} F \text{ then } \mathcal{B}F \ (introspection).$$

Example 4 The program of example 1 translates into T:

$$\begin{array}{l} \mathcal{E}b \Rightarrow \mathcal{E}a \\ \mathcal{E} \sim a \end{array}$$

whose least model is $\{\mathcal{E} \sim a\}$. Thus an agent with knowledge T believes all of $\mathcal{B}\mathcal{E} \sim a$, $\mathcal{B} \sim \mathcal{E}a$, $\mathcal{B} \sim \mathcal{E}b$, and $\mathcal{B} \sim \mathcal{E} \sim b$.

Moreover we insist on the principle that, for rational agents,

$$\text{if } T \models \mathcal{E}L \text{ then } \mathcal{B} \sim \mathcal{E} \sim L \text{ (coherence).}$$

Coherence states that whenever L is provenly true then it is mandatory to believe that L is not provenly false[5]. The *coherence principle* introduced for extended logic programming in [PA92] is an instance of it. In the above example absence of coherence does not interfere with the result. This is not in general the case:

Example 5 Consider $T = \{\mathcal{E}a; \ \mathcal{E} \sim a\}$ whose least model is $\{\mathcal{E}a, \mathcal{E} \sim a\}$. $\mathcal{B}\mathcal{E}a$ and $\mathcal{B}\mathcal{E} \sim a$ hold by introspection. Moreover, by coherence, an agent must sustain both $\mathcal{B} \sim \mathcal{E} \sim a$ and $\mathcal{B} \sim \mathcal{E}a$.

This kind of reasoning may seem strange since the agent must believe in complementary formulae (e.g. in $\mathcal{E}a$ and in $\sim \mathcal{E}a$.). But, as shown below, when the axioms for $\mathcal{B}$ are introduced, we'll see, these will detect inconsistency out from the intuitively inconsistent theory T, i.e. belief cannot be held of proven complements.

As for $\mathcal{E}$, little is assumed about $\mathcal{B}$, both for the sake of flexibility and because it is indeed enough for characterizing *WFSX*. More precisely, we assume the axioms introduced in [Prz93] for the belief operator:

- For any tautologically false formula F: $\sim \mathcal{B}F$.
- For any formulae F and G: $\mathcal{B}(F \wedge G) \equiv \mathcal{B}F \wedge \mathcal{B}G$.

[5] Note that $\mathcal{B} \sim \mathcal{E} \sim L \equiv \mathcal{B}\mathcal{M}L$.

As proven in [Prz93], from these axioms it follows for every formula F that

$$\mathcal{B}F \Rightarrow \sim\mathcal{B} \sim F \ ^{6}$$

. Consequently, from believing two complementary formulae, $\mathcal{B}F$ and $\mathcal{B} \sim F$, inconsistency follows because $\mathcal{B} \sim F \Rightarrow \sim\mathcal{B}F$.

In summary, for a theory T resulting from a definite extended program, the set of beliefs of an agent is the closure, under the above axioms, of:

$$\{\mathcal{B}F \mid T \models_{min} F\} \cup \{\mathcal{B} \sim\mathcal{E} \sim F \mid T \models \mathcal{E}F\}$$

as required by introspection and coherence, respectively.

In order to enlarge the logic **K** with a nonmonotonic belief operator we proceed as above, but now consider the case where formulae of the form $\mathcal{B}F$ or $\sim\mathcal{B}F$ (hereafter called belief formulae) occur in theories. In this case it is not adequate to obtain the belief closure as above. To deal with belief formulae in theories we must consider, as usual in AEL, the expansions of a theory.

An expansion T^* of a theory T is a fixpoint of the equation:

$$T^* = T \cup Bel$$

where Bel is a set of belief formulae depending on T^*. Intuitively, each expansion stands for a belief state of a rational agent. One issue arises: *which kind of nonmonotonicity to introduce in such theories?*

In this respect two main approaches have been followed in the literature:

- One, present in Moore's AEL and in reflexive AEL, is based on CWA – an agent believes in F in an expansion T^* iff $T^* \models F$, and does not believe in F iff $T^* \not\models F$ – and it captures two–valued (or total) logic program semantics, i.e. those where whenever A does not belong to a model then $not\ A$ belongs to it.

- The other approach is based on GCWA – an agent believes in F in an expansion T^* iff $T^* \models_{min} F$, and does not believe in F iff $T^* \models_{min} \sim F$ – and captures three–valued (or partial) logic program semantics. This approach is followed in the AEL of closed beliefs [Prz91a], and in his static semantics [Prz93][7].

Here we adopt the second approach too. The reasons for prefering a logic based on GCWA rather than on CWA are tantamount to those that prefer semantics based on WFS rather than on stable models, and are extensively discussed in the literature (e.g. in [AP92, Bon92, GRS91, Prz91a, Prz93]). In this paper we do not go into the details for this preference, but summarize [Prz91a]: With CWA quite "reasonable"

[6]In fact, this implication is equivalent to $\sim(\mathcal{B}F \wedge \mathcal{B} \sim F)$, by the second axiom it is equivalent to $\sim\mathcal{B}(F \wedge \sim F)$, which is true because $F \wedge \sim F$ is tautologically false.

[7]Note that the question of distinguishing between these two approaches is not relevant for definite programs, since in them nonderivability coincides with deriving the complement in the (single) minimal model.

theories are often inconsistent; expansions are non–cumulative, non–rational, and non–relevant even for theories resulting from normal logic programs[8]; expansions cannot be effectively computed (even for propositional logic programs); the insistance on total models often lacks expressivity. Non of this occurs with GCWA based expansions.

In the sequel we formally define our epistemic logic. We begin by extending the language of propositional logic with modal operators $\mathcal{E}$ and $\mathcal{B}$, standing for "provability" and "belief". Theories are recursively defined as usual. Moreover we assume every theory contains all axioms of logic **K** for $\mathcal{E}$, and the above two axioms for $\mathcal{B}$.

Definition 1.1 *A* minimal model of a theory T *is a model* M *of* T *such that there is no smaller model* N *of* T *coinciding with* M *on belief propositions.*
If F *is true in all minimal models of* T *then we write* $T \models_{min} F$.

An expansion T^* corresponds to a belief state where the agent believes in F if $T^* \models_{min} F$, and does not believe in F if $T^* \models_{min} \sim F$. With the axioms introduced for $\mathcal{B}$, the second statement is subsumed by the first. Indeed, by the first statement, if $T^* \models_{min} \sim F$ then $\mathcal{B} \sim F$, and from the axioms for $\mathcal{B}$ it follows, so we've seen, that $\sim \mathcal{B} F$.

Just as argued for definite extended programs, when considering theories with provability and belief one new form of belief obtention (coherence) is in place, namely if $T^* \models \mathcal{E}G$ then $\mathcal{B} \sim \mathcal{E} \sim G$. Thus, expansions should formalize the following notion of belief $\mathcal{B}$:

$$\mathcal{B}F \quad \equiv \quad F \text{ is minimally entailed, or } F = \sim \mathcal{E} \sim G \text{ and } \mathcal{E}G \text{ is entailed.}$$

Definition 1.2 *An* expansion of a theory T *is a consistent theory* T^* *satisfying the fixed point condition:*

$$T^* = T \cup \{\mathcal{B}F \mid T^* \models_{min} F\} \cup \{\mathcal{B} \sim \mathcal{E} \sim G \mid T^* \models \mathcal{E}G\}$$

Example 6 [9] Consider an agent with the following knowledge:

- Peter is a bachelor;
- a man is not married if he is a bachelor;
- Susan is married to Peter, if we don't believe she's married to Tom;
- Susan is married to Tom, if we don't believe she's married to Peter;
- no one is married to oneself;

[8]By cumulativity [Dix91] we refer to the efficiency related ability of using lemmas. By rationality [Dix91] we refer to the ability to add the negation of a non–provable conclusion without changing the semantics. By relevance [Dix92] we mean that the top–down evaluation of a literal's truth–value requires only the call–graph below it

[9]This example first appeared in [Wag93], in the form of a logic program.

rendered by the autoepistemic theory T (with obvious abbreviations):

$$\begin{aligned} & \mathcal{E}b(p) \\ \mathcal{E}b(X) \Rightarrow & \ \mathcal{E} \sim m(X,Y) \\ \mathcal{B} \sim \mathcal{E}m(t,s) \Rightarrow & \ \mathcal{E}m(p,s) \\ \mathcal{B} \sim \mathcal{E}m(p,s) \Rightarrow & \ \mathcal{E}m(t,s) \\ & \ \mathcal{E} \sim m(X,X) \end{aligned}$$

The only expansion of T contains, among others, the belief propositions:

$$\{\mathcal{B}\mathcal{E}b(p), \mathcal{B}\mathcal{E} \sim m(p,s), \mathcal{B} \sim \mathcal{E}m(p,s), \mathcal{B}\mathcal{E}m(t,s)\}$$

In this example all of an agent's beliefs are completely decided, in the sense that for any proposition A the agent either believes or disbelieves A. This is not in general the case.

Example 7 Consider the statements:

- it is proven or it is believed that the car can be fixed;
- if it is not believed that one can fix the car then an expert is called for;
- an expert is not called for;

rendered by the autoepistemic theory T :

$$\begin{aligned} \mathcal{E}can\_fix\_car \ \vee & \ \mathcal{B}\mathcal{E}can\_fix\_car \\ \mathcal{B} \sim \mathcal{E}can\_fix\_car \Rightarrow & \ \mathcal{E}call\_expert \\ & \ \mathcal{E} \sim call\_expert \end{aligned}$$

The only expansion of T is:

$$T \cup \{\mathcal{B}\mathcal{E} \sim call\_expert, \mathcal{B} \sim \mathcal{E}call\_expert\}$$

stating that an agent believes that an expert is not called and that he disbelieves an expert is called for.

Note that about $\mathcal{E}can\_fix\_car$ the agent remains undefined. This is due to, on the one hand, believing it true is impossible since it is not a consequence in all minimal models; on the other hand, believing it false leads to an inconsistency.

Like Moore's autoepistemic theories $\mathcal{E}\mathcal{B}$ theories might have several expansions:

Example 8 Consider the theory T, describing the so–called Nixon diamond situation:

$$\begin{aligned} & & \mathcal{E}republican(nixon) \\ & & \mathcal{E}quaker(nixon) \\ \mathcal{E}republican(X), \mathcal{B} \sim \mathcal{E}pacifist(X) & \Rightarrow & \mathcal{E} \sim pacifist(X) \\ \mathcal{E}quaker(X), \mathcal{B} \sim \mathcal{E} \sim pacifist(X) & \Rightarrow & \mathcal{E}pacifist(X) \end{aligned}$$

T has three expansions, namely:

$$
\begin{array}{ll}
T \cup \{\mathcal{BE}r(n), \mathcal{BE}q(n), \mathcal{BE}p(n), & \mathcal{B} \sim\mathcal{E} \sim r(n), \mathcal{B} \sim\mathcal{E} \sim q(n), \mathcal{B} \sim\mathcal{E} \sim p(n)\} \\
T \cup \{\mathcal{BE}r(n), \mathcal{BE}q(n), \mathcal{BE} \sim p(n), & \mathcal{B} \sim\mathcal{E} \sim r(n), \mathcal{B} \sim\mathcal{E} \sim q(n), \mathcal{B} \sim\mathcal{E}p(n)\} \\
T \cup \{\mathcal{BE}r(n), \mathcal{BE}q(n), & \mathcal{B} \sim\mathcal{E} \sim r(n), \mathcal{B} \sim\mathcal{E} \sim q(n)\}
\end{array}
$$

The first states that it is believed that Nixon is a pacisfist; the second that it is believed that Nixon is not a pacifist; and the third remains undefined in what concerns Nixon being a pacisfist or not.

When confronted with several expansions (i.e. several possible states of beliefs) a sceptical reasoner should only conclude what is common to all. Here that means the third expansion.

1.3 Relation to extended logic programs

An extended logic program is as set of rules of the form:

$$L_0 \leftarrow L_1, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_n \tag{3}$$

where each L_i is an objective literal, i.e. an atom A or its $\neg$–negation $\neg A$.

As argued above, an atom A is translated into $\mathcal{E}A$, and an explicitly negated atom $\neg A$ into $\mathcal{E} \sim A$. In [LS93b, MT93] literals of the form $not\ L$ (default literals) are translated into $\mathcal{L} \sim \mathcal{L}L$ in reflexive AEL. [MT93] gives an intuitive reading of this formula:"it is known that L is not known". In our approach we translate $not\ L$ into $\mathcal{B} \sim\mathcal{E}L$, i.e. "it is believed (or it is assumed) that L is not proven". So, each rule of the form (3) is translated into:

$$\mathcal{E}L_1, \ldots, \mathcal{E}L_m, \mathcal{B} \sim\mathcal{E}L_{m+1}, \ldots, \mathcal{B} \sim\mathcal{E}L_n \Rightarrow \mathcal{E}L_0 \tag{4}$$

Definition 1.3 *A* WFSX *partial stable model M of an extended logic program P corresponds to an expansion T^* when:*

- *For an objective literal L : $L \in M$ iff $\mathcal{BE}L \in T^*$.*
- *For a default literal $not\ L$: $not\ L \in M$ iff $\mathcal{B} \sim\mathcal{E}L \in T^*$.*

Theorem 1.2 *Let T be the theory obtained from an extended logic program P by means of translation (4). Then there is a one–to–one correspondence between the* WFSX *partial stable models of P and the expansions of T.*

This relationship brings mutual benefits to both *WFSX* and the $\mathcal{EB}$ logic. On the one hand, the logic allows for a more intuitive view of *WFSX*, specially in what concerns its understanding as modeling provability and belief in a rational agent. This allows for a clearer formulation within *WFSX* of some problems in knowledge representation and reasoning, and for a better understanding of *WFSX*'s results. In particular, it shows that explicit negation stands for proving falsity of a literal, default negation for believing that a literal is not provable, and undefinedness for believing neither the falsity nor the verity of a literal. The relationship also sheds light on several extensions of *WFSX* (cf. section 3).

On the other hand, for the class of theories resulting from some extended programs, the logic can be implemented using the top–down procedures defined for *WFSX*[ADP94]. Moreover, for this class, the logic enjoys the properties of cumulativity, rationality, relevance [Dix92], and others proven for *WFSX* in [Alf93]. In addition, the relationship also raises new issues in epistemic logics, and points towards their solution via the techniques in use in extended logic programming (cf. section 3).

2 Provability versus Knowledge

Above we claimed logic $\mathcal{EB}$ is flexible and general. Next we express with it different meanings for $\mathcal{E}$, and hence a variety of semantics for extended logic programs.

The logic **K** introduced for $\mathcal{E}$ is the simplest normal modal system, contained in any other. With additional axioms to our theories we can define other meanings for $\mathcal{E}$. In particular, with the axioms of logic **SW5**[10] $\mathcal{E}$ represents "knowledge", as in [Sch91, MT93]. Other formalizations of knowledge, such as that of logic **S4.2**[11], are similarly obtained.

Using the **SW5** meaning of $\mathcal{E}$, but keeping with the same translation, a different semantics for extended logic programs is obtained:

Theorem 2.1 *Let T be the theory obtained from an extended logic program P by means of translation (4), augmented with the* **SW5** *axioms for $\mathcal{E}$. Then there is a one–to–one correspondence between expansions of T and the partial stable models of P of WFS with strong negation, as defined in [AP92].*

Comparisons between WFS with strong negation and *WFSX*, found in [AP92, Alf93], prove the former is less suitable for implementation (as it does not enjoy relevance [Dix92]), is more credulous, and assigns semantics to less programs.

Example 9 Program P :

$$\begin{array}{rcl} \neg a & & \\ a & \leftarrow & not\, b \\ b & \leftarrow & not\, b \end{array}$$

translates into the theory:

$$T = \{\mathcal{E} \sim a;\ \ \mathcal{B} \sim \mathcal{E}b \Rightarrow \mathcal{E}a;\ \ \mathcal{B} \sim \mathcal{E}b \Rightarrow \mathcal{E}b\}.$$

Using logic **K**, there is one expansion:

$$T^* = T \cup \{\mathcal{BE} \sim a, \mathcal{B} \sim \mathcal{E}a, \mathcal{B} \sim \mathcal{E} \sim b\}.$$

[10] I.e. axioms T: $\mathcal{E}F \Rightarrow F$, 4: $\mathcal{E}F \Rightarrow \mathcal{EE}F$, and W5: $\sim\mathcal{E} \sim F \Rightarrow (F \Rightarrow \mathcal{E}F)$.

[11] [LS93a] uses **S4.2** to formalize knowledge in a logic which also includes belief. We intend to compare this logic with ours when their final version becomes available to us.

If logic **SW5** is used instead, there is no expansion. This happens because, by axiom T, $\mathcal{E} \sim a$ entails $\sim\mathcal{E}a$, and by the contrapositive of the second clause of T $\sim\mathcal{E}a$ entails $\sim\mathcal{B} \sim\mathcal{E}b$. Thus, by the third clause, every minimal model of every possible expansion has $\sim\mathcal{E}b$, and so $\mathcal{B} \sim\mathcal{E}b$ must be added. This is inconsistent with having $\sim\mathcal{B} \sim\mathcal{E}b$ in all models, and so no expansion exists. *WFSX* assigns a meaning to P, namely $\{\neg a, not\ a, not\ \neg b\}$, because axiom T is not assumed.

From theorem 2.1 and the results of [AP92] it follows that:

Theorem 2.2 *Let T be the theory obtained from an extended logic program P by means of translation (4), augmented with the* **SW5** *axioms for $\mathcal{E}$, and the axiom $\sim\mathcal{E}F \Rightarrow \mathcal{E} \sim F$. Then there is a one–to–one correspondence between the expansions of T and the partial stable models of P of the "stationary semantics with classical negation" of [Prz91b].*

Since answers–sets are the total stable models of WFS with strong negation [AP92]:

Definition 2.1 *An expansion T^* is total iff for every formula F :*

$$T^* \not\models \mathcal{B}F \quad \Rightarrow \quad T^* \models \mathcal{B}\sim F$$

Theorem 2.3 *Let T be the theory obtained from an extended logic program P by means of translation (4), augmented with the* **SW5** *axioms for $\mathcal{E}$. Then there is a one–to–one correspondence between the total expansions of T and the answer–sets of P.*

3 Further developments

Since the language of $\mathcal{EB}$ is more general than that of extended logic programs, our logic is a tool for further generalizations of extended logic programming, for instance disjunction. All that is required is to define a translation of disjunctive extended logic programs into the logic. The study of possible translations, and the relationship between the resulting and extant semantics for disjunctive programs is the subject of ongoing investigations by us.

Another possible direct generalization of extended logic programming is with the modal operators of the logic, allowing for conjunction and disjunction within their scope. Examples of the use and usefulness of the belief operator for normal disjunctive programs can be found in [Prz93].

With the relationship between $\mathcal{EB}$ logic and extended logic programming now established some issues already tackled in the latter can also be raised in the former. Furthermore, the former can profit from adapting techniques employed in the latter. One of the issues presented here in more detail is contradiction removal, or belief revision.

Recently, several authors have studied this issue in extended logic programming [DR91, Jon91, PA93, PAA91a]. The basic idea behind these approaches is that $not\ L$

literals be viewed as assumptions, so that if an assumption partakes in a contradiction then its revision is in order. In epistemic logics this idea translates into: "If the results of introspection lead to the inexistence of expansions then revise your beliefs".

Example 10 The theory T:

$$\begin{array}{rcl} \mathcal{B} \sim\mathcal{E}ab & \Rightarrow & \mathcal{E}fly \\ & & \mathcal{E} \sim fly \end{array}$$

is consistent but has no expansion. This is so because $\sim\mathcal{E}ab$ is true in all minimal models and thus, by introspection, $\mathcal{B} \sim\mathcal{E}ab$ must be added causing a contradiction. In fact, this is a typical case where the result of introspection leads to contradiction[12].

In order to assign a meaning to consistent theories without expansions two approaches are possible: to define a more sceptical notion of expansion, introducing less belief propositions by introspection; or to minimally revise the theory in order to provide for expansions. [AP94] contrast these two approaches in the logic programming setting, dubbing the first "contradiction avoidance", and the second "contradiction removal", and showing them equivalent under certain conditions. The techniques used there to deal with this issue in logic programming can readily be employed in our epistemic logic:

Contradiction avoidance in the $\mathcal{EB}$ logic amounts to weakening the condition for introspection. This can be accomplished by allowing introduction of belief propositions solely for some chosen subset of the formulae minimally entailed by the theory. Of course, not all subsets are allowed. In particular, we are only interested in maximal subsets compatible with consistency. The study of additional preference conditions among these subsets is tantamount to the one in extended logic programming examined in [PA93].

Contradiction removal in the $\mathcal{EB}$ logic amounts to minimally adjoining, to some consistent theory without expansions, new clauses to inhibit the addition by introspection of belief propositions responsible for contradiction.

Example 11 The theory T' resulting from adjoining the "inhibiting clause"

$$\mathcal{B} \sim\mathcal{E}ab \Rightarrow \mathcal{E}ab$$

to the theory of example 10 has one expansion

$$T^* = T' \cup \{\mathcal{BE} \sim fly, \mathcal{B} \sim\mathcal{E}fly\}.$$

The extra clause states that believing $\sim\mathcal{E}ab$ is impossible, since it directly implies $\mathcal{E}ab$, and in this way inhibits the otherwise inevitable addition of $\mathcal{B} \sim\mathcal{E}ab$.

In extended logic programs the inhibition clause above translates to

$$ab \leftarrow not\ ab$$

[12] Note this problem is not peculiar to our logic. The same occurs as well in e.g. AEL and reflexive AEL.

and is called an inhibition rule for ab [PAA91a]. Similarly to what is done in [PAA91a] for extended logic programming, also for the $\mathcal{EB}$ logic we can define the revisions of a theory as those obtained by minimal additions of inhibiting clauses which achieve contradiction removal (i.e. allow the resulting theory to have an expansion). Thus, the revisions of a theory T are those theories

$$T' = T \cup InR$$

where InR is a minimal set of clauses of the form $\mathcal{B} \sim\mathcal{E}A \Rightarrow \mathcal{E}A$, such that T' has at least one expansion. Given the similarities of this approach and the contradiction removal techniques in extended logic programs, the procedures [AP94] and implementations of the latter, developed in order to avoid generating all possible revisions, are also applicable to the former.

Further discussion, and results, on this topic can be found in a recent paper by us an Teodor Przymusinski [APP94].

Conclusions

To start, we've provided a first embedding, in a two-modality AEL, of well–founded semantics based extensions of logic programs, either with explicit or strong negation [AP92], and shown how they differ in that setting.

Second, we've contrasted this embedding with recent ones for stable semantics based extensions of logic programs with a second kind of negation [Che93, LS93b, MT93, Pea93].

Third, we've shown the usefulness of employing the combination of the two natural epistemic modalities of provability and belief for explicating the semantics of logic programs in AEL.

Fourth, we've introduced into epistemic logics the issue of belief revision in the face of inexistance of expansions, and how to tackle it inspired by similar questions and attending techniques previously developed, in the logic programmming context, for contradiction removal [Jon91, PAA91a, AP94].

Fifth, we've shown how (via the procedures and implementations of the extended logic programs we've embedded in our AEL) an important subset of this AEL can be employed to represent and solve a variety of nonmonotonic reasoning problems, already captured in and dealt with by extended logic programs: namely taxonomies, reasoning about actions, diagnosis, updates, and debugging [PAA91b, PAA93, PDA93].

Sixth, as the language of our epistemic logic is more general than that of extended programs, the former can now be used as a tool for further generalizations of the latter, for instance wrt disjunction and modalities.

Acknowledgments

We acknowledge Esprit BRA Compulog 2 (no. 6810), and JNICT - Portugal for their support. Thanks to Carlos Damásio and Teodor Przymusinski for helpful discussions.

References

[ADP93] J. J. Alferes, P. M. Dung, and L. M. Pereira. Scenario semantics of extended logic programs. In L. M. Pereira and A. Nerode, editors, *2nd Int. Ws. on LP & NMR*, pages 334–348. MIT Press, 1993.

[ADP94] J. J. Alferes, C. V. Damásio, and L. M. Pereira. Top-down query evaluation for well-founded semantics with explicit negation. In A. Cohn, editor, *European Conf. on AI*, pages 140–144. Morgan Kaufmann, 1994.

[Alf93] José Júlio Alferes. *Semantics of Logic Programs with Explicit Negation*. PhD thesis, Universidade Nova de Lisboa, October 1993.

[AP92] J. J. Alferes and L. M. Pereira. On logic program semantics with two kinds of negation. In K. Apt, editor, *Int. Joint Conf. and Symp. on LP*, pages 574–588. MIT Press, 1992.

[AP94] J. J. Alferes and L. M. Pereira. Contradiction: when avoidance equal removal. In R. Dyckhoff, editor, *4th Int. Ws. on Extensions of LP*, volume 798 of *LNAI*. Springer–Verlag, 1994.

[APP94] J. J. Alferes, L. M. Pereira, and T. Przymusinski. Belief revision in non-monotonic reasoning and logic programming. Technical report, CRIA, UNINOVA and Univ. of California at Riverside, December 1994.

[BF87] N. Bidoit and C. Froidevaux. Minimalism subsumes default logic and circumscription in stratified logic programming. In *Symp. on Principles of Database Systems*. ACM SIGACT-SIGMOD, 1987.

[Bon92] P. Bonatti. Autoepistemic logics as a unifying framework for the semantics of logic programs. In K. Apt, editor, *Int. Joint Conf. and Symp. on LP*, pages 417–430. MIT Press, 1992.

[Che80] B. Chellas. *Modal Logic: An introduction*. Cambridge Univ. Press, 1980.

[Che93] J. Chen. Minimal knowledge + negation as failure = only knowing (sometimes). In L. M. Pereira and A. Nerode, editors, *2nd Int. Ws. on LP & NMR*, pages 132–150. MIT Press, 1993.

[Dix91] J. Dix. Classifying semantics of logic programs. In A. Nerode, W. Marek, and V. S. Subrahmanian, editors, *LP & NMR*, pages 166–180. MIT Press, 1991.

[Dix92] J. Dix. A framework for representing and characterizing semantics of logic programs. In B. Nebel, C. Rich, and W. Swartout, editors, *3rd Int. Conf. on Principles of Knowledge Representation and Reasoning*. Morgan Kaufmann, 1992.

[DR91] P. M. Dung and P. Ruamviboonsuk. Well founded reasoning with classical negation. In A. Nerode, W. Marek, and V. S. Subrahmanian, editors, *LP & NMR*, pages 120–132. MIT Press, 1991.

[Dun91] P. M. Dung. Negation as hypotheses: An abductive framework for logic programming. In K. Furukawa, editor, *8th Int. Conf. on LP*, pages 3–17. MIT Press, 1991.

[EK89] K. Eshghi and R. Kowalski. Abduction compared with negation by failure. In *6th Int. Conf. on LP*. MIT Press, 1989.

[Gel87] M. Gelfond. On stratified autoepistemic theories. In *AAAI'87*, pages 207–211. Morgan Kaufmann, 1987.

[GL88] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In R. Kowalski and K. A. Bowen, editors, *5th Int. Conf. on LP*, pages 1070–1080. MIT Press, 1988.

[GL90] M. Gelfond and V. Lifschitz. Logic programs with classical negation. In Warren and Szeredi, editors, *7th Int. Conf. on LP*, pages 579–597. MIT Press, 1990.

[GRS91] A. Van Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991.

[HC84] G. Hughes and M. Cresswell. *A companion to modal logic*. Methuen, 1984.

[Jon91] K. Jonker. On the semantics of conflit resolution in truth maintenance systems. Technical report, Univ. of Utrecht, 1991.

[KS90] R. Kowalski and F. Sadri. Logic programs with exceptions. In Warren and Szeredi, editors, *7th Int. Conf. on LP*. MIT Press, 1990.

[Lif92] V. Lifschitz. Minimal belief and negation as failure. Technical report, Dep. of Computer Science and Dep. of Philisophy, Univ. of Texas at Austin, 1992.

[LS93a] P. Lamarre and Y. Shoham. On knowledge, certainty, and belief (draft). Personal communication of the second author, Stanford Univ., 1993.

[LS93b] V. Lifschitz and G. Schwarz. Extended logic programs as autoepistemic theories. In L. M. Pereira and A. Nerode, editors, *2nd Int. Ws. on LP & NMR*, pages 101–114. MIT Press, 1993.

[Min87] J. Minker. On indefinite databases and the closed world assumption. In M. Ginsberg, editor, *Readings in Nonmonotonic Reasoning*, pages 326–333. Morgan Kaufmann, 1987.

[Moo85] R. Moore. Semantics considerations on nonmonotonic logic. *Artificial Intelligence*, 25:75–94, 1985.

[MT93] V. Marek and M. Truszczynski. Reflexive autoepistemic logic and logic programming. In L. M. Pereira and A. Nerode, editors, *2nd Int. Ws. on LP & NMR*, pages 115–131. MIT Press, 1993.

[Nel49] D. Nelson. Constructible falsity. *JSL*, 14:16–26, 1949.

[PA92] L. M. Pereira and J. J. Alferes. Well founded semantics for logic programs with explicit negation. In B. Neumann, editor, *European Conf. on AI*, pages 102–106. John Wiley & Sons, 1992.

[PA93] L. M. Pereira and J. J. Alferes. Optative reasoning with scenario semantics. In D. S. Warren, editor, *10th Int. Conf. on LP*, pages 601–615. MIT Press, 1993.

[PAA91a] L. M. Pereira, J. J. Alferes, and J. N. Aparício. Contradiction Removal within Well Founded Semantics. In A. Nerode, W. Marek, and V. S. Subrahmanian, editors, *LP & NMR*, pages 105–119. MIT Press, 1991.

[PAA91b] L. M. Pereira, J. N. Aparício, and J. J. Alferes. Nonmonotonic reasoning with well founded semantics. In Koichi Furukawa, editor, *8th Int. Conf. on LP*, pages 475–489. MIT Press, 1991.

[PAA92] L. M. Pereira, J. J. Alferes, and J. N. Aparício. Default theory for well founded semantics with explicit negation. In D. Pearce and G. Wagner, editors, *Logics in AI. Proceedings of the European Ws. JELIA'92*, volume 633 of *LNAI*, pages 339–356. Springer–Verlag, 1992.

[PAA93] L. M. Pereira, J. N. Aparício, and J. J. Alferes. Non–monotonic reasoning with logic programming. *Journal of Logic Programming. Special issue on Nonmonotonic reasoning*, 17(2, 3 & 4):227–263, November 1993.

[PDA93] L. M. Pereira, C. Damásio, and J. J. Alferes. Diagnosis and debugging as contradiction removal. In L. M. Pereira and A. Nerode, editors, *2nd Int. Ws. on LP & NMR*, pages 316–330. MIT Press, 1993.

[Pea93] D. Pearce. Answer sets and constructive logic, II: Extended logic programs and related nonmonotonic formalisms. In L. M. Pereira and A. Nerode, editors, *2nd Int. Ws. on LP & NMR*, pages 457–475. MIT Press, 1993.

[PP90] H. Przymusinska and T. Przymusinski. Semantic issues in deductive databases and logic programs. In R. Banerji, editor, *Formal Techniques in AI, a Sourcebook*, pages 321–367. North Holland, 1990.

[Prz90] T. Przymusinski. Extended stable semantics for normal and disjunctive programs. In Warren and Szeredi, editors, *7th Int. Conf. on LP*, pages 459–477. MIT Press, 1990.

[Prz91a] T. Przymusinski. Autoepistemic logic of closed beliefs and logic programming. In A. Nerode, W. Marek, and V. S. Subrahmanian, editors, *LP & NMR*, pages 3–20. MIT Press, 1991.

[Prz91b] T. Przymusinski. A semantics for disjunctive logic programs. In Loveland, Lobo, and Rajasekar, editors, *ILPS'91 Ws. in Disjunctive Logic Programs*, 1991.

[Prz93] T. Przymusinski. Static semantics for normal and disjunctive programs. Technical report, Dep. of Computer Science, Univ. of California at Riverside, 1993.

[Rei78] R. Reiter. On closed–world data bases. In H. Gallaire and J. Minker, editors, *Logic and DataBases*, pages 55–76. Plenum Press, 1978.

[Rei80] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:68–93, 1980.

[Sch91] G. Schwarz. Autoepistemic logic of knowledge. In A. Nerode, W. Marek, and V. S. Subrahmanian, editors, *LP & NMR*, pages 260–274. MIT Press, 1991.

[Wag91] G. Wagner. A database needs two kinds of negation. In B. Thalheim, J. Demetrovics, and H-D. Gerhardt, editors, *Mathematical Foundations of Database Systems*, volume 495 of *LNCS*, pages 357–371. Springer–Verlag, 1991.

[Wag93] G. Wagner. Reasoning with inconsistency in extended deductive databases. In L. M. Pereira and A. Nerode, editors, *2nd Int. Ws. on LP & NMR*, pages 300–315. MIT Press, 1993.

A Answer-sets, and *WFSX* definitions

An extended program is a set of rules of the form:

$$L_0 \leftarrow L_1, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_n\ (0 \leq m \leq n)$$

where each L_i is an objective literal. An objective literal is either an atom A or its explicit negation $\neg A$. We also use $\neg$ to denote complementary literal wrt. the explicit negation, so that $\neg\neg A = A$. The set of all objective literals of a program P is called the extended Herbrand base of P and denoted by $\mathcal{H}(P)$. The symbol not stands for negation by default. $not\ L$ is called a default literal. Literals are either objective or default literals. By $not\ \{a_1, \ldots, a_n, \ldots\}$ we mean $\{not\ a_1, \ldots, not\ a_n, \ldots\}$. An interpretation of an extended program P is denoted by $T \cup not\ F$, where T and F are disjoint subsets of $\mathcal{H}(P)$. Objective literals in T are said to be *true* in I, objective literals in F *false by default* in I, and in $\mathcal{H}(P) - I$ *undefined* in I.

We begin by recalling the definition of answer-sets semantics:

Definition A.1 (The Γ–operator) *Let P be an extended program, T a set of objective literals, and let P' (resp. T') be obtained from P (resp. T) by denoting every literal $\neg A$ by a new atom, say $\neg\_A$. The GL–transformation $\frac{P'}{T'}$ is the program obtained from P' by removing all rules containing a default literal $not\ A$ such that $A \in T'$, and by then removing all the remaining default literals from P. Let J be the least model of $\frac{P'}{T'}$. $\Gamma_P T$ is obtained from J by replacing the introduced atoms $\neg\_A$ by $\neg A$.*

Definition A.2 (Answer-sets semantics) *Let P be an extended program. An interpretation $I = T \cup not\ F$ is an answer-set of P iff:*

$$T = \Gamma_P T \quad \textit{and} \quad F = \mathcal{H}(P) - T$$

The answer-sets semantics of P is determined by the intersection of all answer-sets of P.

For similarity with the definition of answer-sets semantics, here we present *WFSX* in a distinctly different manner with respect to its original definition. This presentation is based on alternating fix-points of Gelfond–Lifschitz Γ–like operators [GL88, GL90]. The proof of equivalence between both definitions, as well as proofs of other results in this section, can be found in [Alf93].

To impose the coherence requirement in WFS we introduce:

Definition A.3 (Semi-normal version of a program) *The semi-normal version of a program P is the program P_s obtained from P by adding to the (possibly empty) $Body$ of each rule $L \leftarrow Body$ the default literal $not\ \neg L$, where $\neg L$ is the complement of L wrt. explicit negation.*

Below we use $\Gamma(S)$ to denote $\Gamma_P(S)$, and $\Gamma_s(S)$ to denote $\Gamma_{P_s}(S)$.

Definition A.4 (Partial stable model) *A set of objective literals T generates a partial stable model (PSM) of an extended program P iff:*

$$\begin{array}{ll} (1) & T = \Gamma\Gamma_s T; \textit{ and} \\ (2) & T \subseteq \Gamma_s T. \end{array}$$

The partial stable model generated by T is the interpretation:

$$T \cup not\ (\mathcal{H}(P) - \Gamma_s T)$$

In other words, partial stable models are determined by the fix-points of $\Gamma\Gamma_s$. Given a fix-point T, objective literals in T are *true* in the PSM, objective literals not in $\Gamma_s T$ are *false by default*, and all others are *undefined*. Thus, objective literals in $\Gamma_s T$ are all the true or undefined ones. Note that condition (2) imposes that a literal cannot be both true and false by default (viz. if it belongs to T it does not belong to $\mathcal{H}(P) - \Gamma_s T$, and vice-versa). Moreover note how the use of Γ_s imposes coherence: if $\neg L$ is true, i.e. it belongs to T, then in $\Gamma_s T$, via semi-normality, all rules for L are removed and, consequently, $L \notin \Gamma_s T$, i.e. L is false by default.

Example 12 Program $P = \{a; \quad \neg a\}$ has no partial stable models. Indeed, the only fix-point of $\Gamma\Gamma_s$ is $\{a, \neg a\}$, and $\{a, \neg a\} \not\subseteq \Gamma_s\{a, \neg a\} = \{\}$. Thus it is not a PSM.

Programs without partial stable models are said *contradictory*.

Theorem A.1 (*WFSX* semantics) *Every non-contradictory program P has a least (wrt. $\subseteq$) partial stable model, the well-founded model of P ($WFM(P)$).*

To obtain an iterative "bottom-up" definition for $WFM(P)$ we define the following transfinite sequence $\{I_\alpha\}$:

$$\begin{array}{rcl} I_0 & = & \{\} \\ I_{\alpha+1} & = & \Gamma\Gamma_s I_\alpha \\ I_\delta & = & \bigcup\{I_\alpha \mid \alpha < \delta\} \quad \textit{for limit ordinal } \delta \end{array}$$

There exists a smallest ordinal λ for the sequence above, such that I_λ is the smallest fix-point of $\Gamma\Gamma_s$, and $WFM(P) = I_\lambda \cup not\ (\mathcal{H}(P) - \Gamma_s I_\lambda)$.

In this constructive definition literals obtained after an application of $\Gamma\Gamma_s$ (i.e. in some I_α) are true in $WFM(P)$, and literals not obtained after an application of Γ_s (i.e. not in $\Gamma_s I_\alpha$, for some α) are false by default in $WFM(P)$.

Theorem A.2 (Relation to WFS) *For normal logic programs (i.e. without explicit negation)* WFSX *coincides with the well–founded semantics of [GRS91].*

Theorem A.3 (Relation to Answer-sets) *Let P be an extended logic program with at least one answer-set. Every answer-set of P is also a PSM of P. Moreover, for any objective literal L:*

- *If $L \in WFM(P)$ then L belongs to all answer-sets of P.*
- *If $not\ L \in WFM(P)$ then L does not belong to any answer-set of P.*