The Extended Stable Models of Contradiction Removal Semantics

Luís Moniz Pereira José J. Alferes
Joaquim N. Aparício
AI Centre, Uninova and DCS, U. Nova de Lisboa
2825 Monte da Caparica
Portugal

March 12, 1992

Abstract

Our purpose is to define a semantics that extends Contradiction Removal Semantics just as Extended Stable Model Semantics extends Well Founded Semantics, thus providing the notion of Contradiction Free Extended Stable Models. Contradiction Removal Semantics extends Well Founded Semantics to deal with contradictions arising from the introduction of classical negation. Because the Extended Stable Models structure of a program is useful for expressing defaults and abduction, it is important to study in what way the structure of the Extended Stable Models is affected by Contradiction Removal Semantics when the Well Founded Model is contradictory. Given that the Contradiction Removal Semantics is useful for expressing belief revision and counterfactual reasoning, dealing with the structure of such models is expected to be useful for mixing together all four mentioned kinds of reasoning within a single common framework.

The Contradiction Free Extended Stable Semantics is an extension to the definition of Contradiction Removal Semantics, provided here in a form independent from the Well Founded Semantics, in terms of a fixpoint operator, instead of in terms of the Well Founded Semantics of transformed programs as before. This definition also clarifies how the semantics is useful for Belief Revision.

Nevertheless, we also define a single program transformation such that the Extended Stable Models of a transformed program correspond exactly to the Contradiction Free Extended Stable Models of the original one. By relying on the Well Founded Semantics of the transformed program no new model determining procedures are needed.

Introduction

Well Founded Semantics (WFS) (introduced in [VGRS90]) is a 3-valued semantics extending previous logic program semantics to the class of all normal programs. This semantics has been proven equivalent to a natural extension to 3 values of the 2-valued Stable Model

Semantics of [GL88], i.e. the 3-valued (or extended) Stable Model Semantics (XSMS) of [PP90, Prz90]. The WF Model (WFM) of a program P is the smallest XS Model of P. In [Prz90], an extension of WFS encompassing programs with classical negation is proposed, similar to the extension of [GL90] to deal with classical negation within Stable Model Semantics. This approach transforms a program with classical negation into another without it, by replacing every occurrence of each classically negated literal by a positive literal with a new predicate symbol. Afterwards the program models are obtained as usual. Finally, every contradictory model[1] is rejected. This seems quite reasonable for noncontradictory programs, i.e. those whose WFM is noncontradictory. But for contradictory ones it seems too strong to throw out all models. Consider for example the statements: *Birds, not known to be abnormal, fly.Tweety is a bird and does not fly. Socrates is a man.* which can be naturally represented by the program[2]:

$$
\begin{array}{lll}
fly(X) \leftarrow bird(X), \sim abnormal(X). & & bird(tweety) \\
\neg fly(tweety). & & man(socrates).
\end{array}
$$

The [Prz90] WFS approach to classical negation provides no model for this program, which in our opinion is inadequate. We should at least be able to say that *Socrates* is a *man*. In our view, it is also reasonable to conclude that *tweety* is a *bird* and doesn't fly, because the rule stating it doesn't fly, being a fact, makes a stronger statement than the one concluding it $flies$, since the latter relies on the assumption of non-abnormality enforced by the Closed World Assumption (CWA) treatment of the negation as failure involving the abnormality predicate. In fact, whenever an assumption leads to a contradiction it seems logical to be able to take it back in order to remove the contradiction.

More recently, in [PAA91a], a semantics was defined that extends WFS to programs with classical negation, which avoids the absence of models caused by contradictions brought about by closed world assumptions – the Contradiction Removal Semantics (CRS). This extension relies on allowing to take back such contradiction originating CWA assumptions about literals, by making their truth value become undefined rather than false, and thus permiting noncontradictory models to appear. Such assumptions can be withdrawn in a minimal way, in all alternative ways of removing contradictions. Moreover, a single unique model that defines the semantics of a program is identified. This model is included in all the alternative contradiction removing ones. Since CRS extends WFS to deal with contradictions arising from the introduction of classical negation, and since the XSM structure of a program is useful for expressing defaults and abduction [PAA91f, PAA91e], it is important to study in what way the structure of the XSMs is affected by CRS when the WFM is contradictory and redefined by CRS as the CRWFM. Given that the CRS is useful for expressing belief revision and counterfactual reasoning [PAA91c], dealing with such models is expected to be useful for mixing together all these four kinds of reasoning within a single common framework.

The definition of these models is an extension to the definition of CRS, here provided in a form independent of the WFS, in terms of a fixpoint operator, instead of in terms of the WFS of transformed programs as before. This definition also clarifies how the semantics is useful for Belief Revision.

[1] By contradictory model we mean one that has some literal p and its classical corresponding negative literal $\neg p$ both true in it. A contradictory program is one having a contradictory WF Model.

[2] Here, and throughout the paper, $\sim$stands for the negation as failure and $\neg$ for classical negation.

Nevertheless, we also define a single program transformation such that the XSMs of a transformed program correspond exactly to the Contradiction Free Extended Stable Models (CFXSMs) of the original one. By relying on the WFS of the transformed program no new model determining procedures are needed; they can be found in [Prz89, War89, PAA90, PAA91d].

The structure of this paper is as follows: in section 1 we define the language used and a program transformation similar to the ones in [GL90] and [Prz90] for dealing with classical negation, but also adumbrating integrity constraints. In section 2 we define some useful sets for establishing the causes of and the removal of contradictions within WFS. Afterwards we define Contradiction Free Extended Stable Models based on a fixpoint operator. In section 4 we formalize a notion of families of CRXSMs and examine some problems of ordering between models and families. Section 5 states some important links with the WFS. Finally, in section 6, we define the single program transformation refered above.

1 Language

In what follows a program is a set of rules of the form:

$$H \leftarrow B_1, \ldots, B_n, {\sim}C_1, \ldots, {\sim}C_m. \quad m \geq 0, n \geq 0.$$

The symbol $\sim$ stands for negation as failure, i.e. negation in the sense of WFS. $H, B_1, \ldots, B_n$, $C_1, \ldots, C_m$ are classical literals. A classical literal is either an atom A or its classical negation $\neg A$. A literal is a classical literal L or its negation as failure ${\sim}L$.

As in [GL90] and in [Prz90], we first transform such programs into ones obtained by replacing every occurrence of every classically negated literal, say $\neg l(X)$, by another with the same arguments and a new predicate name, say '$\neg l$'(X). Then we introduce, for every atom $a(X)$ of the program, the integrity constraint $\perp \leftarrow a(X),$'$\neg a$'(X), where $\perp$ stands for not true[3].

After the above transformation, contradiction removal is tantamount to preventing the appearance of literal $\perp$ in the WFM. This notion of contradiction can be extended to encompass the prevention of integrity constraint (IC) violation, if each IC is of the form $\leftarrow A_1, \ldots, A_n$ and transformed into the rule $\perp \leftarrow A_1, \ldots, A_n$.

Throughout the paper we consider partial (or 3-valued) interpretations. A partial interpretation [PP90] I of a language $\mathcal{L}$ is a pair $< T; F >$ where T and F are disjoint subsets of the Herbrand base $\mathcal{H}$ of $\mathcal{L}$. The set T contains all ground atoms true in I, the set F contains all ground atoms false in I and the truth value of the remaining atoms, those in $U = \mathcal{H} - (T \cup F)$, is undefined(or unknown).

We represent a partial interpretations $< T; F >$, equivalently, by a set of literals I such that:

- $A \in I$ iff $A \in T$
- ${\sim}A \in I$ iff $A \in F$

[3] Because, according to WFS, a literal can only be true if it has at least one rule defined for it, we need only add these $\perp$-rules for every pair of atoms $a(X)$ and $\neg a(X)$ that figure (both) as conclusions of rules in the program.

By contradictory interpretation we mean one that has $\perp$ true in it. A contradictory program is one having a contradictory WF Model.

2 Contradiction Support Sets and Contradiction Removal Sets

In this section we present the (assumptive) Contradiction Support Sets and (assumptive) Contradiction Removal Sets which were first defined in [PAA91a]. These sets are the main new constructs required by Contradiction Removal Semantics.

Informally, Contradiction Support Sets are sets of $\sim$negative literals present in the WF Model which are sufficient to lend support to $\perp$ in the WFM (and thus support a contradiction)[4], i.e. given their truth the truth of $\perp$ is inevitable. Contradiction Removal Sets are built from the Contradiction Support Sets. Intuitively, they are minimal sets of literals chosen from the Support Sets such that any support of $\perp$ registrers at least one literal in the set. Consequently, if all literals in some Contradiction Removal Set were to become undefined in value no support of $\perp$ would exist. We shall see how such literals can be made undefined, through revising a contradictory program by means of a transformation.

Example 1 Consider the program:

$$p \leftarrow \sim q. \qquad p \leftarrow \sim r. \qquad \neg p \leftarrow \sim t. \qquad a \leftarrow \sim b.$$

Its contradiction support sets are $\{\sim q, \sim t\}$ and $\{\sim r, \sim t\}$, and its contradiction removal sets are $\{\sim q, \sim r\}$ and $\{\sim t\}$.

Suppose we had q and r both undefined. In that case $\perp$ would also be undefined, the program becoming noncontradictory. The same would happen if t alone became undefined. No other set, not containing one of these two alternatives, has this property. $\square$

Definition 2.1 *Support Set*[5]

A Support Set of a literal L belonging to the WF Model M_P of a program P, represented as $SS_P(L)$, or $SS(L)$ for short, is obtained as follows:

- *If L is an atom:*
  - *Choose some rule of P for L where all the literals in its body belong to M_P. One $SS(L)$ is obtained by taking all those body literals plus the literals in some SS of each body literal.*
- *If $L = \sim A$:*
  - *If there are no rules defined for A in P then the only SS of L is $\{L\}$.*

[4] This notion can be seen as a special case of the notion of Suspect Sets, both of wrong and missing solutions, in declarative debugging [PC88, PCA90].

[5] An alternative definition of support sets [PAA90] relies on a notion of derivation for a literal in the WFS, and doesn't require the previous availability of the WF Model.

– *Otherwise, choose from each rule defined for A, a literal such that its complement*[6] *belongs to M_P. A $SS(L)$ has all those complement literals, and the literals of a SS of each of them.*

By considering all possible rules of P for a literal all its SSs are obtained.

This definition of Support Set in the WFM, can easily be extended to XSMs. For this we have only to consider the model M_P above as being some XSM. In this case, we dub the sets, *Extended Support Sets* (XSS).

Example 2 Consider the program:

$$p \leftarrow \sim q, r. \qquad p \leftarrow \sim b. \qquad r \leftarrow a. \qquad b \leftarrow q, c. \qquad a.$$

whose WF Model is $M_P = \{p, r, a, \sim q, \sim b, \sim c\}$. Let's compute the SSs of p.

By the first rule for p, because $\sim q, r \in M_P$ they belong to a $SS(p)$. As there are no rules for q the only SS of $\sim q$ is $\{\sim q\}$. From the single rule for r we get that the single SS of r is $\{a\}$, since the only SS of a is $\{\}$. So one $SS(p) = \{\sim q, r, a\}$. With the second rule for p we get the only other $SS(p) = \{\sim b, \sim q, \sim c\}$. $\Box$

Proposition 2.1 *Existence of Support Sets*

Every literal L belonging to the WF Model of a program P has at least one support set $SS_P(L)$.

Proof: The proof follows from the definition of WFM as in [PP90], and is omitted for brevity. $\diamondsuit$

We have a special interest in those negative literals true by CWA, i.e. those for each there are no rules defined for their complement. With the purpose of identifying such literals we define the Assumption Support Set of a literal in the WF Model.

Definition 2.2 *Assumption Support Set*

An Assumption Support Set (ASS) of a literal L in the WFM is the (possibly empty) subset of some $SS_P(L)$, which contains exactly all its $\sim$negative elements having no rules for their complement. We represent an ASS of a literal L in a program P as $ASS_P(L)$, or $ASS(L)$ for short. For simplicity we represent these sets using the literal's complements (i.e. with atoms).

Example 3 The two $ASS(p)$ in example 2 are $\{q\}$ and $\{q, c\}$. $\Box$

Definition 2.3 $\perp$*-Assumption Support Set*

A $\perp$-Assumption Support Set ($\perp ASS$) is an $ASS(\perp)$.

These are sets of atoms false by CWA in the WF Model of the program, involved in supporting contradiction (i.e. $\perp$) in the program[7].

Having defined the sets of CWA literals that together support some literal, it is easy to produce sets of CWA literals such that, if all become undefined, the truth of that literal necessarily becomes ungrounded.

[6]The complement literal of an atom A is $\sim A$; that of a literal of the form $\sim A$ is A.

[7]Note that there is a close relationship between the SSs of $\perp$ and the sets of *NOGOODS* of Truth Maintenance Systems.

Definition 2.4 *Removal Sets*

A Removal Set (RS) of a literal L belonging to the WFM of a program P is a set of atoms formed by the union of some nonempty subset from each $ASS_P(L)$. If the empty set is an $ASS_P(L)$, then the only $RS(L)$ is the empty set. Note that a literal not belonging to the WFM of P has no RSs defined for it.

In view of considering minimal changes to the WF Model, we next define those *RS*s which are minimal in the sense that there is no other *RS* contained in them.

Definition 2.5 *Minimal RS*

A RS of a literal L, $RS_m(L)$, in a program P, is minimal iff there exists no $RS_i(L)$ in P such that $RS_m(L) \supset RS_i(L)$. We represent a minimal RS of L in P as $MRS_P(L)$.

Definition 2.6 *Contradiction Removal Sets*

A Contradiction Removal Set (CRS) of program P is a minimal Removal Set of the (special) literal $\perp$, i.e. a CRS of P is a $MRS_P(\perp)$.

3 Contradiction Free Extended Stable Models

Next we define Combined Removal Sets (CombRSs), based on the CRSs, to obtain the contradiction removal 3-valued stable models (CFXSMs). The CombRSs are the sets of literals which, if made undefined, generate the noncontradictory models of interest (i.e. the CFXSMs).

Definition 3.1 *Combined Removal Sets*

Let $Lits_{CRS}$ be the set of all atoms in some CRS of a program P, i.e. $Lits_{CRS} = \bigcup_i CRS_i$. A Combined Removal Set (CombRS) of P is a set of atoms formed from the union of a CRS with a subset of $Lits_{CRS}$. If there are no CRSs for P then, by definition, its only Combined Removal Set is the empty set.

This notion is an obvious extension to the notion of CRS. It gives all the ways of removing contradiction, considering the literals that contribute to it, i.e. those that contain some CRS. CRSs are minimal CombRSs.

CFXSMs will be defined in such a way that in each CFXSM the literals of some CombRS are all undefined. This ensures that the models are noncontradictory. For this purpose we define the program transformation:

Definition 3.2 *Given a program P and a set A of atoms from its alphabet, we define P_A, and call it P sub A, as the program obtained by:*

- *Removing from P all rules whose head is an element of A.*
- *Replacing every occurrence of each atom A in P by the special atom* $\mathbf{u}$[8].

[8]The meaning of this special atom, introduced in [PP90], is that $\mathbf{u}$ is undefined in every model of any program.

The alphabet of P_A, $\mathcal{LIT}(P_A)$, is by definition, $\mathcal{LIT}(P) - \mathcal{LIT}(A)$.

Note that no XSM of the transformed program contains any element in A (i.e. all are undefined). However, because of the second item above, all the consequences of forcing them undefined are nevertheless obtained.

We can now define Contradiction Free Extended Stable Models (CFXSMs). The idea is that the CFXSMs of each program P are the XSMs of a program P_R obtained by the above transformation on P with respect to each CombRS R. The desired result is that the set of CFXSMs of a program P is the set of all noncontradictory XSMs of each P_R, when every CombRS is considered. More formally:

Definition 3.3 *Modulo transformation [PP90]*

Let P be a logic program and let I be a 3-valued interpretation. By the extended GL-transformation of P modulo I we mean a new (non-negative) program P/I obtained from P by performing the following three operations:

- *Removing from P all rules which contain a negative premise $L = \sim A$ such that $A \in I$.*
- *Removing from all the remaining rules those negative $L = \sim A$ which satisfy $L \in I$.*
- *Replacing in all remaining rules the negative premises $L = \sim A$ by $\mathbf{u}$.*

$\Gamma^*(P, I)$, a generalization of the Γ operator [GL88], is defined as the 3-valued least model of P/I.

Definition 3.4 *Contradiction Free Extended Stable Models*

A noncontradictory partial interpretation I is a Contradiction Free Extended Stable Model of a program P iff there exists some CombRS R of P such that $I = \Gamma^*(P_R, I)$. If such is the case we call R the Source Removal Set of I.

The need to impose noncontradiction on the XSMs is that literals undefined in it, even if the WFM is noncontradictory, still may originate an XSM of some transformed program P_{RS} with some contradiction, when they become defined. This matter will be further examined later.

Example 4 Consider the (contradictory) program $P = \{p \leftarrow \sim q;\ \neg p \leftarrow \sim r;\ a \leftarrow \sim b;\ b \leftarrow \sim a\}$, whose (contradictory) WF Model is $\{p, \neg p, \sim q, \sim r\}$. Its CRSs are $\{q\}$ and $\{r\}$, and its CombRSs are its CRSs and $\{q, r\}$.

The corresponding transformed programs are:

$$\begin{aligned} P_{\{q\}} &= \{p \leftarrow \mathbf{u};\ \neg p \leftarrow \sim r;\ a \leftarrow \sim b;\ b \leftarrow \sim a\} \\ P_{\{r\}} &= \{p \leftarrow \sim q;\ \neg p \leftarrow \mathbf{u};\ a \leftarrow \sim b;\ b \leftarrow \sim a\} \\ P_{\{q,r\}} &= \{p \leftarrow \mathbf{u};\ \neg p \leftarrow \mathbf{u};\ a \leftarrow \sim b;\ b \leftarrow \sim a\} \end{aligned}$$

For the first program we have the CFXSMs: $\{\neg p, \sim r\}$, $\{\neg p, \sim r, a, \sim b\}$ and $\{\neg p, \sim r, b, \sim a\}$. For the second program we have the CFXSMs: $\{p, \sim q\}$, $\{p, \sim q, a, \sim b\}$ and $\{p, \sim q, b, \sim a\}$. Finally, for the last program we have the CFXSMs: $\{\}$, $\{a, \sim b\}$ and $\{b, \sim a\}$.

These are the only CFXSMs. The first group of three correspond to the XSMs of the program, after removing the contradiction via undefining q. The second group correspond to the XSMs of the program, after removing the contradiction via undefining r. As there is no compulsory preference about which literal to remove, q or r, we should also consider undefining them both. In this case the modified XSMs of the program are those in the third group. □

Lemma 3.1 *If the empty set is a CRS of a program P then P has no CFXSMs.*

Proof: If the empty set is a CRS of P then there exists an empty $\perp$-ASS, i.e. there exists one $SS(\perp)$ having no CWAs, and all CombRSs are $\{\}$. Since $P = P_{\{\}}$ has one $SS(\perp)$ having no CWAs then its WFM is contradictory. So all its XSM are contradictory, i.e. non of them is a CFXSM. ♢

Lemma 3.2 *If the empty set is not a CRS of a program P then there exists at least one CFXSM of P.*

Proof: By hypothesis, program P has no empty CRS. Two cases occur: if there are no CRSs the conclusion follows by theorem 5.1; otherwise there is at least one nonempty CRS, RS. By definition of Contradiction Removal Set, none of the previous $SS(\perp)$ exists in P_{RS}, because all atoms in RS are undefined in P_{RS}, and hence in all its XSMs[9]. Moreover as undefined literals cannot make any other literals true or false, no new $SS(\perp)$ are generated. ♢

With the two lemmas above, the next theorem, which expresses the completeness conditions for the semantics, follows easily.

Theorem 3.3 *A program P has at least one (noncontradictory) CFXSM iff the empty set is not a CRS of P.*

4 Ordering among CFXSMs

In this section we formalize the notion of family of CFXSMs and establish orderings among models and their families. In order to take advantage of previous results, we compare the orderings with the lattice of submodels of [PAA91a]. We also identify a single unique model that defines the semantics of a program. This model is proven equivalent to the one defined in [PAA91a].

Definition 4.1 *Two CFXSMs of a program P belong to the same family $[RS]$ if they have the same Source Removal Set RS.*

Intuitively, a family corresponds to the set of XSMs of the resulting program after one way of removing the contradiction.

Example 5 In example 4, each of the three groups of CFXSMs is one family. □

[9] One always exists, albeit the WFM of P_{RS}.

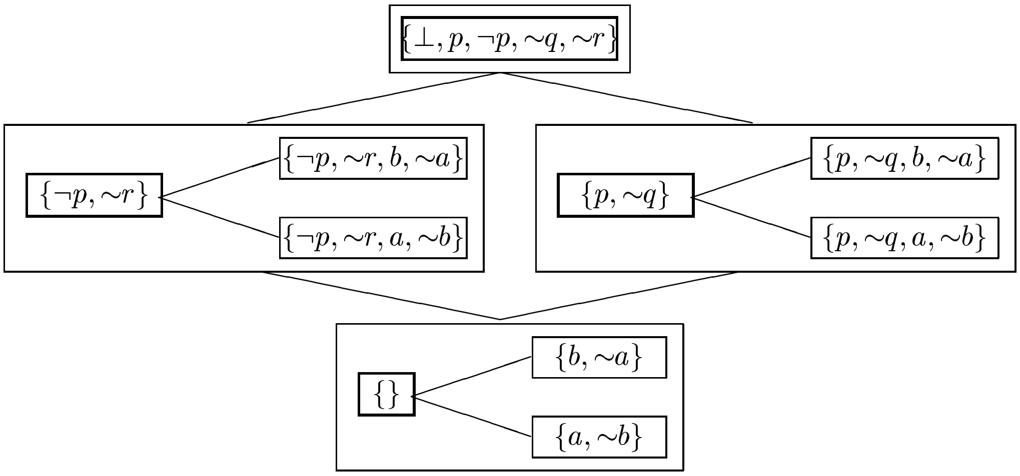
A set inclusion ordering among models of the same family can be defined: We say a model $M_1 \leq M_2$ iff $M_1 \subseteq M_2$. Given this ordering, we define the root model $root([RS])$ of a family $[RS]$ to be its least model, i.e. by the properties of XSM Semantics $root([RS])$ is the WFM of P_{RS}.

Theorem 4.1 *If a program P has at least one CFXSM then for every CombRS RS there exists a nonempty family $[RS]$.*

Proof: By theorem 3.3, this is the case when the empty set is not a CRS of P. In all (possible contradictory) XSMs of P_{RS} all atoms in RS are undefined. By definition of Contradiction Removal Set, in every $SS(\bot)$ of P, at least one element was removed (i.e. became undefined) in P_{RS}. So none of the $SS(\bot)$ remains in P_{RS}. As undefined literals cannot make any other literal true or false, no new $SS(\bot)$ are generated. Thus at least the WFM of P_{RS} is noncontradictory and so is a CFXSM with Source Removal Set RS. $\diamondsuit$

Definition 4.2 *We say $[RS_1] \leq [RS_2]$ iff $root([RS_1]) \subseteq root([RS_2])$. In order to ensure this set inclusion ordering is a lattice we include as top element the family comprised of the (contradictory) WFM of the original program.*

Example 6 The families of example 5 and their ordering is represented by the lattice below.



where families are $\leq$ from bottom to top, and models within a family are $\leq$ from left to right. $\square$

A family reflects the XSMs structure of the original P after a specific contradiction removal. Its root model is its WFM. This suggests a close relation between root models and submodels (from [PAA91a]). In fact:

Proposition 4.1 *The root model of each family $[RS]$ of a program P is a submodel of P.*

Proof: $root([RS])$ is the WFM of P_{RS}, and P_{RS} is the program obtained from P after forcing literals in RS undefined. Because all atoms in RS are CWA literals, by definition root models are submodels. $\diamondsuit$

Proposition 4.2 *A root model of a maximal noncontradictory family (i.e. maximal family excluding the top element) is a maximal noncontradictory submodel, and vice-versa.*

Proof: The proof follows easily, given that the set of literals that became undefined are the same in both cases, i.e. the CRSs. $\diamondsuit$

Theorem 4.2 *Relation with Contradiction Removal Semantics*
The Contradiction Removal Well Founded Model (as defined in [PAA91a]) is the root model, $root([RS])$, of the least family RS.

By establishing the connection with the definitions in [PAA91a], the notions of minimality of change defined there are inherited by the CFXSMs.

5 Relations with WFS

In this section will relate the CFXSMs with the (possibly contradictory) XSMs of any program[10]. In particular we prove that for noncontradictory programs they coincide.

The next result expresses why the new semantics is an extension of XSM Semantics.

Theorem 5.1 *If there exists one noncontradictory XSM for a program P, then an interpretation I is a noncontradictory XSM of P iff it is a CFXSM of P.*

Proof: If there exists one noncontradictory XSM for P, then the WFM of P does not contain $\perp$. Then by proposition 2.1 P has no CRSs. In this case the only CombRS is $\{\}$. So by definition 3.4, the CFXSMs are the noncontradictory XSMs of $P_{\{\}}$ and as $P_{\{\}} = P$ by definition, they are also the noncontradictory XSMs of P. $\diamondsuit$

In fact this theorem expresses that whenever the XSM Semantics provides a noncontradictory model, the new semantics also provides the same models and vice-versa. Moreover, in cases where the XSM Semantics of [Prz90] does not provide any semantics, the new semantics does provide one, which consists not only of a unique noncontradictory model, the CRWFM, but also of some more CFXSMs.

In the general case the program may be contradictory, and the relationship between the models of the XSM Semantics and those of the CFXSM Semantics is established via the families in the latter.

A family can be viewed as the XSM structure pertaining to one way of removing contradiction. The relationship between CFXSMs in families and the (contradictory) XSMs of the original program, allows to establish results on the changes to the XSMs after contradiction removal. The CFXSMs can then be put to use for abductive, counterfactual, and default reasoning, as in [PAA91f, PAA91e, PAA91c].

We next formalize the (partial) mapping from a given contradictory XSM CX of a program P, whose contradictory WFM is M_P, to a XSM FX of some family $[RS]$.

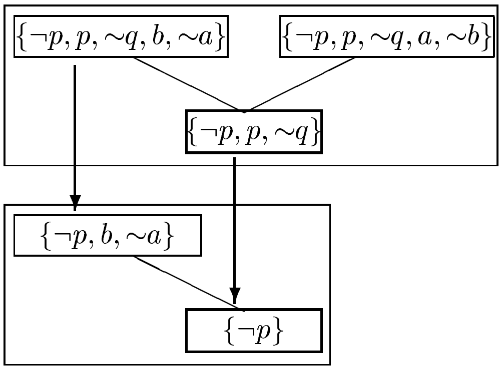
Let $Sup = CX - M_P$. If such a mapping exists the intended result is that FX results from $root([RS])$ in the same way as CX results from M_P (i.e. by defining the same set

[10] In what follows we consider that contradictory programs can have XSMs, not giving, for this purpose any special meaning to the atom $\perp$.

Sup of atoms[11]). In other words $CX - M_P = FX - root([RS])$, and so that CX maps into $FX = (CX - M_P) \cup root([RS])$.

Compared with CX, in FX (if it exists) there are some atoms that become undefined. The mapping is not defined if for some literal l in Sup all its Extended Support Sets $XSS(l)$ contain the complement of some atom in RS: in that case there could be no support for l in FX, and FX would not contain all of Sup.

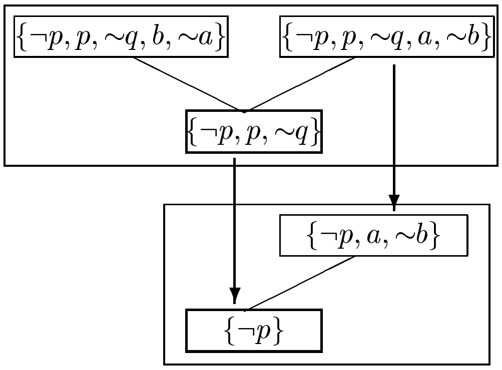
Example 7 Consider the contradictory program $\{p \leftarrow \sim q; \quad \neg p; \quad b \leftarrow \sim a; \quad a \leftarrow \sim b; \sim q\}$. Its only CombRS is $\{q\}$. The XSM structure of the program and its only family $[\{q\}]$ are:



where the arrows express the desired partial mapping; note there is no mapping for the rightmost model because the only $XSS(a)$[12] in it contains $\sim q$, which becomes undefined in the family since $q \in RS$. $\square$

The only case when a CX does not have a mapping is when it would map into a contradictory FX. This occurs only when there exists a $XSS(\perp)$ containing some element of Sup but not any complement literal of an atom in RS, i.e. such a contradiction is not removed by undefining the elements in RS (cf. formal definition below).

Example 8 Consider the contradictory program $\{p \leftarrow \sim q; \quad \neg p; \quad p \leftarrow \sim a; \quad a \leftarrow \sim b; \quad b \leftarrow \sim a\}$. Its only CombRS is $\{q\}$. The XSM structure of the program and the only family $[\{q\}]$ of the latter are:



There is no mapping for the leftmost model because there is a $XSS(\perp) = \{\neg p, p, \sim a\}$ which persists even after making q undefined. $\square$

[11] The literals in Sup are undefined in the WFM, thus not appearing in any SS and consequently in any CombRS.

[12] Note that in this case a is a member of Sup.

Definition 5.1 *Given a program P with WFM M_P and a family $[RS]$, we define the partial mapping $\xi : XSM \longrightarrow CFXSM$ such that $\xi(M_P, RS)(CX) = (CX - M_P) \cup root([RS])$ only if both two conditions below hold:*

- $\forall l \in Sup,\ \exists XSS(l) \mid \widehat{RS} \cap XSS(l) = \{\}$
- $\forall XSS(\bot),\ Sup \cap XSS(\bot) = \{\}\ \vee \widehat{RS} \cap XSS(\bot) \neq \{\}$

where $Sup = CX - M_P$ and $\widehat{RS}$ is the set of complement of the atoms in RS.

Proof of the theorem below is omitted here for brevity, but appears in [PAA91b].

Theorem 5.2 *Correctness of the mapping*

An element belongs to the family $[RS]$ of a program P with WFM M_P iff it belongs to the codomain of $\xi(M_P, RS)$.

Example 9 Consider the program:

$$
\begin{array}{lllllllll}
\neg p. & & & a & \leftarrow & \sim b, \sim q. & c & \leftarrow & \sim d. \\
p & \leftarrow & b, c. & b & \leftarrow & \sim a. & d & \leftarrow & \sim c. \\
p & \leftarrow & \sim q. & & & & & &
\end{array}
$$

whose only CRS is $\{q\}$. Its XSMs are:

$$
\begin{array}{cl}
\{\neg p, p, \sim q\} & WFM \\
\{\neg p, p, \sim q, a, \sim b\} & XSM_1 \\
\{\neg p, p, \sim q, b, \sim a\} & XSM_2 \\
\{\neg p, p, \sim q, c, \sim d\} & XSM_3 \\
\{\neg p, p, \sim q, d, \sim c\} & XSM_4 \\
\{\neg p, p, \sim q, a, \sim b, c, \sim d\} & XSM_5 \\
\{\neg p, p, \sim q, a, \sim b, d, \sim c\} & XSM_6 \\
\{\neg p, p, \sim q, b, \sim a, c, \sim d\} & XSM_7 \\
\{\neg p, p, \sim q, b, \sim a, d, \sim c\} & XSM_8
\end{array}
$$

Let's construct now the CFXSMs of the only family $[\{q\}]$ of P.

Its root is the WFM of $P_{\{q\}}$ which is $\{\neg p\}$.

The mapping does not apply to XSM_1, XSM_5 and XSM_6, since they violate the first condition: all of them have a, and all supports of a fall when q becames undefined.

It also does not apply to XSM_7, since it violates the second condition: b and c in that model cause a contradiction not removed by undefining q.

Thus the CFXSMs of $[\{q\}]$ are:

$$
\begin{array}{cc}
\{\neg p\} & root([\{q\}]) \\
\{\neg p, b, \sim a\} & \textit{by applying } \xi \textit{ to } XSM_2 \\
\{\neg p, c, \sim d\} & \textit{by applying } \xi \textit{ to } XSM_3 \\
\{\neg p, d, \sim c\} & \textit{by applying } \xi \textit{ to } XSM_4 \\
\{\neg p, b, \sim a, d, \sim c\} & \textit{by applying } \xi \textit{ to } XSM_8
\end{array}
$$

□

6 A single transformation giving all CFXSMs

Now we present an equivalent alternative definition of CR Semantics, based on a single transformation of any program P, that not only gives us the CRWFM but also the several CFXSMs. The WF Model of the transformed program is the CRWF Model of the original one. The other noncontradictory XSMs of the transformed program are the CFXSMs.

As we've seen in section 3, each CFXSM coresponds to one possible way of removing contradiction, by undefining at least all the elements of some CRS and optionally, additional elements of other CRSs. In order to make available the option of undefining or not some CWA-false literal of P, we can add to P, for that CWA literal's atom A, the rules $A \leftarrow \sim A'$ and $A' \leftarrow \sim A$, where A' is a new atom with the arguments of A. These rules have the effect of making A (and A') undefined in the WF Model, and of allowing XSMs with A (and $\sim A'$) and others with $\sim A$ (and A'). To prevent A being true in any XSM, we prevent A' being false in any model by also adding the rule $A' \leftarrow \sim A'$.

Definition 6.1 *CWA choice rules*

The CWA choice rules for a classical literal A are $A \leftarrow \sim A'$, $A' \leftarrow \sim A$ and $A' \leftarrow \sim A'$, where A' is a new atom with the arguments of A.

Proposition 6.1 *Let P be a program without rules defined for atom A. The truth value of A in the WF Model of P augmented with the CWA choice rules for A is undefined. There exists at least one XSM of the augmented program such that $\sim A$ belongs to it. There is no such XSM of the augmented program such that A belongs to it.*

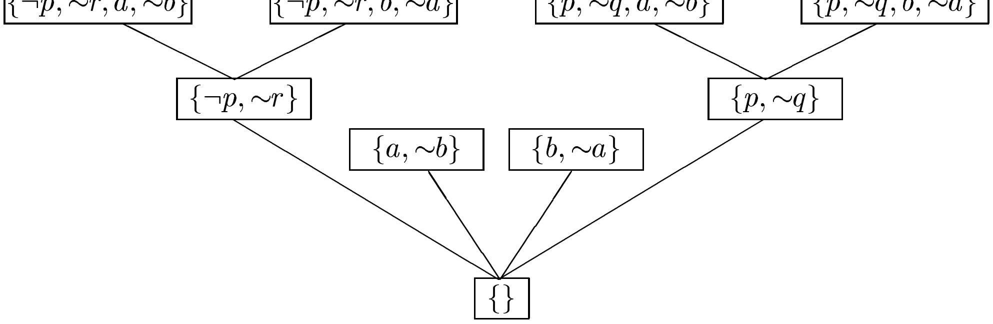
Definition 6.2 *CR Program Transformation*

The CR Transformed Program of a contradictory program $CRTP(P)$ with CRSs $C_1, \ldots, C_n$ is the program obtained by adding to P all CWA choice rules for each element of each C_i $(1 \leq i \leq n)$[13]*.*

Example 10 The CRTP of the program of example 4 is:

$$
\begin{array}{rcl@{\qquad}rcl@{\qquad}rcl}
p & \leftarrow & \sim q. & q & \leftarrow & \sim q' & r & \leftarrow & r' \\
\neg p & \leftarrow & \sim r. & q' & \leftarrow & \sim q & r' & \leftarrow & r \\
a & \leftarrow & \sim b. & q' & \leftarrow & \sim q' & r' & \leftarrow & r' \\
b & \leftarrow & \sim a. & & & & & &
\end{array}
$$

and its XSMs (modulo r' and q'), are shown in the next figure, where its contradictory models have been discarded:



[13] Note that the union of all CRSs is equal to the union of all CombRSs.

Note that they correspond exactly to the CFXSMs found in example 4. □

To prove the correctness of this transformation, let's first state some lemmas.

Lemma 6.1 *There is a bijection between the noncontradictory XSM of $CRTP(P)$ (modulo new atoms introduced by CWA choice rules) and the CFXSMs of P.*

Proof: We prove this lemma by constructing for a given XSM of $CRTP(P)$ XM, a CombRS RS such that XM is also an XSM of P_{RS}, and thus a CFXSM.

By proposition 6.1, atoms in some subset S of $Lits_{CRS}$(c.f. definition 3.1) of P are false in XM, and atoms in $Lits_{CRS} - S$ are undefined in XM. Let $RS = Lits_{CRS} - S$. All XSMs of P_{RS} have, as noted before, all atoms in RS undefined and, as there are no rules for them, all atoms in S false. Thus, with respect to atoms of $Lits_{CRS}$, all XSMs of P_{RS} are equal to XM. The P sub RS transformation does not affect other literals, so if there is stability for them in XM, the same applies in one XSM of P_{RS}. $\diamondsuit$

Lemma 6.2 *If $[RS]$ is a family of a contradictory program P, then every model $M \in [RS]$ is a noncontradictory XSM of $CRTP(P)$ (modulo new atoms introduced by CWA choice rules).*

Proof: The proof of this lemma is similar to the one of lemma 6.1, but this time constructing the various XMs given RS. $\diamondsuit$

Theorem 6.3 *Correctness of the transformation*

An interpretation I is a CFXSM of a program P iff I (modulo new atoms introduced by CWA choice rules) is a noncontradictory XSM of $CRTP(P)$.

Proof: When the CR Semantics coincides with the WFS, i.e. under the conditions of theorem 5.1, there are no CRSs, and so $CRTP(P) = P$. In this case the result follows easily. For a contradictory program the result follows directly from lemmas 6.1 and 6.2 above. $\diamondsuit$

Corollary 1 *If the WFM of $CRTP(P)$ is noncontradictory then it coincides with the CRWFM of P.*

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

[GL88] 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.

[GL90] M. Gelfond and V. Lifschitz. Logic programs with classical negation. In *ICLP'90*, pages 579–597, 1990.

[PAA90] L. M. Pereira, J. N. Aparício, and J. J. Alferes. Top-Down procedures for well founded semantics. Technical report, AI Centre/Uninova, 1990.

[PAA91a] L. M. Pereira, J. J. Alferes, and J. N. Aparício. Contradiction removal within Well Founded Semantics. In *LPNMR'91*. MIT Press, 1991.

[PAA91b] L. M. Pereira, J. J. Alferes, and J. N. Aparício. The extended stable models of contradiction removal semantics. Technical report, AI Centre/Uninova, 1991.

[PAA91c] L. M. Pereira, J. N. Aparício, and J. J. Alferes. Counterfactual reasoning based on revising assumptions. Technical report, AI Centre/Uninova, 1991.

[PAA91d] L. M. Pereira, J. N. Aparício, and J. J. Alferes. A derivation procedure for extended stable models. In *IJCAI'91*. Morgan Kaufmann Publishers, 1991.

[PAA91e] L. M. Pereira, J. N. Aparício, and J. J. Alferes. Hypothetical reasoning with well founded semantics. In *SCAI'91*. IOS Press, 1991.

[PAA91f] L. M. Pereira, J. N. Aparício, and J. J. Alferes. Nonmonotonic reasoning with well founded semantics. In *ICLP'91*. MIT Press, 1991.

[PC88] L. M. Pereira and M. Calejo. A framework for Prolog debugging. In K. B. R. Kowalski, editor, *ICLP'88*. MIT Press, 1988.

[PCA90] L. M. Pereira, M. Calejo, and J. N. Aparício. Refining knowledge base updates. In *Simpósio Brasileiro de Inteligência Artificial*, 1990.

[PP90] H. Przymusinska and T. Przymusinski. *Semantic Issues in Deductive Databases and Logic Programs*. Formal Techniques in Artificial Intelligence. North Holland, 1990.

[Prz89] T. Przymusinski. Every logic program has a natural stratification and an iterated fixed point model. In *8th Symposium on Principles of Database Systems*. ACM SIGACT-SIGMOD, 1989.

[Prz90] T. Przymusinski. Extended stable semantics for normal and disjunctive programs. In *ICLP'90*, pages 459–477. MIT Press, 1990.

[VGRS90] 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.

[War89] D.S. Warren. The XWAM: A machine that integrats prolog and deductive databases. Technical report, SUNY at Stony Brook, 1989.