# Computing Stable and Partial Stable Models of Extended Disjunctive Logic Programs *

Carolina Ruiz Jack Minker

Institute for Advanced Computer Studies and
Department of Computer Science
University of Maryland. College Park, MD 20742 U. S. A.
{cruizc, minker}@cs.umd.edu

**Abstract**

In [Prz91], Przymusinski introduced the partial (or 3-valued) stable model semantics which extends the (2-valued) stable model semantics defined originally by Gelfond and Lifschitz [GL88]. In this paper we describe a procedure to compute the collection of all partial stable models of an extended disjunctive logic program. This procedure consists in transforming an extended disjunctive logic program into a constrained disjunctive program free of negation-by-default whose set of 2-valued minimal models corresponds to the set of partial stable models of the original program.

## 1 Introduction

The partial (or 3-valued) stable model semantics defined by Przymusinski in [Prz91] is a three-valued semantics for the class of extended disjunctive logic programs (*edlps*). This class of programs consists of disjunctive logic programs that may contain two kinds of negations: negation-by-default and explicit negation. The definition of this semantics extends the (2-valued) stable model semantics given by Gelfond and Lifschitz ([GL88]) to the 3-valued disjunctive case.

The original definitions of both the 2-valued and the 3-valued stable model semantics are not constructive. They give criteria to check whether or not a given model of the program is (partial) stable. Some procedures to compute the 2-valued stable model semantics of disjunctive logic programs have been described ([BNNS93, FLMS93, IKH92]).

The purpose of this paper is to provide a procedure that constructs the collection of 3-valued stable models of any *edlp*. To prove that our procedure

*Support for this paper was provided by the Air Force Office of Scientific Research under grant number 91-0350, and the National Science Foundation under grant numbers IRI-8916059 and IRI 9300691.

is correct, we introduce a new characterization of the partial stable model semantics in terms of well-supported 3-valued models of *edlps*. The notion of well-supported 2-valued models was introduced by Fages ([Fag91]) for the class of definite normal programs. Here we extend that notion to *edlps* and to the 3-valued case. As stated by Fages, well-supported models are supported models with loop-free finite justifications. We show that the notions of partial stability and 3-valued well-supportedness are equivalent. This result generalizes Fages work to the 3-valued disjunctive framework. To prove this characterization, we introduce a fixpoint operator that computes the minimal (with respect to the *truth ordering*) 3-valued models of an *edlp* free of negation-by-default.

It is worth noticing that even for the propositional case, the problem of constructing the collection of partial stable models of an *edlp* is not tractable. [1] This is a consequence of the fact that skeptical reasoning in this semantics (i.e. determining if a literal is *true* in every partial stable model of the program) is $\Pi_2^P$–complete (see [EG93]).

Our construction of the collection of partial stable models of a given *edlp* $P$ is as follows: first $P$ is translated into a new constrained *edlp*, called $P^{3S}$, free of negation-by-default whose syntax captures the well-supported semantics of $P$, in the sense that $P^{3S}$ contains clauses stating explicitly when there is support for an atom to be *true*, *false* or *unknown*. Furthermore, constraints appearing in the clauses are used to guarantee that those supports are loop-free. Subsequently, the minimal 2-valued models of $P^{3S}$ are computed. These models, when translated to the language of $P$, are precisely the well-supported (and hence the partial stable) models of $P$.

This paper is organized as follows: Section 2 presents background on the partial stable model semantics needed in the following sections. Section 3 provides both a characterization of partial stable models as well-supported 3-valued models and a fixpoint operator that computes the minimal (with respect to the *truth ordering*) 3-valued models of *edlps* free of negation-by-default. Section 4 is concerned with the computation of the 3-valued stable models of an *edlp*. We introduce a transformation, called the *3S–transformation*, that, given an *edlp* $P$, computes a constrained *edlp* $P^{3S}$. We prove that there is a one-to-one correspondence between the minimal 2-valued models of $P^{3S}$ and the 3-valued well-supported models (and consequently the 3-valued stable models) of the original program. An algorithm to compute the minimal 2-valued models of $P^{3S}$ is given in section 4.2. In section 5 we draw some conclusions.

## 2 Background

Classical logic assumes that the truth value of every sentence is either *true* or *false*. 3-valued semantics allow the additional possibility that the truth value of a statement is *unknown*. In this section we make precise what an *edlp* is and define the notions of 3-valued interpretation and 3-valued model of an *edlp*. We describe alternative orderings on the three truth values and study the orderings

[1] Assuming that P $\neq$ NP.

among 3-valued interpretations that they induce. Finally, the set of 3-valued stable models of an *edlp* is defined.

**Definition 2.1 (Extended disjunctive logic programs).**
Let $\mathcal{L}$ denote a first order language.

1. An *extended disjunctive clause* is a clause of the form:

$$l_0 \vee \ldots \vee l_k \leftarrow l_{k+1}, \ldots, l_m, not\ l_{m+1}, \ldots, not\ l_n$$

   where $0 \leq k \leq m \leq n$ and the $l$'s are literals (i.e. atoms and explicitly negated atoms) in the language $\mathcal{L}$ and *not* is the negation-by-default operator.

2. An *extended disjunctive logic program* (*edlp*) is a set of extended disjunctive clauses.

In what follows we sometimes abbreviate an extended disjunctive clause of the form $l_0 \vee \ldots \vee l_k \leftarrow l_{k+1}, \ldots, l_m, not\ l_{m+1}, \ldots, not\ l_n$ as $H \leftarrow B$ where $H = l_0 \vee \ldots \vee l_k$ and $B = l_{k+1}, \ldots, l_m, not\ l_{m+1}, \ldots, not\ l_n$.

Since an *edlp* is equivalent to the set of all its ground instances, we consider here only propositional *edlps*, and so the language $\mathcal{L}$ is just a set of propositional symbols. We require that $\mathcal{L}$ contain special propositions $\mathbf{t}$, $\mathbf{f}$ and $\mathbf{u}$, that are intended to denote *true, false* and *unknown*, respectively.

Minker and Ruiz ([MR93, MR94]) give techniques to obtain the semantics of an *edlp* in term of the semantics of a corresponding *edlp* free of explicit negation. Therefore, without loss of generality we consider in the sequel only programs free of explicit negation. With this in mind, we say that an *edlp* is *positive* when it is free of negation-by-default.

**Definition 2.2 (Ordering among truth values).**
Consider the following orderings among truth values:

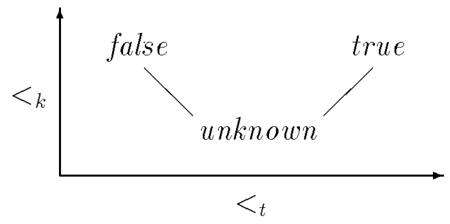
1. *Truth Ordering* ($<_t$) *on truth values*:

$$\textit{false} <_t \textit{unknown} <_t \textit{true}.$$

2. *Knowledge Ordering* ($<_k$) *on truth values*:

$$\textit{unknown} <_k \textit{false} \text{ and } \textit{unknown} <_k \textit{true}.$$

Graphically,



Given a propositional language $\mathcal{L}$, a 3-valued interpretation is a 3-valued truth assignment to the propositions in $\mathcal{L}$. It is commonly represented as a partial function (hence the name of *partial* interpretation) $I : \mathcal{L} \longrightarrow \{true, false\}$ in which the truth value of a proposition that does not belong to the domain of $I$ is taken to be *unknown*. A concise way of writing such a partial function is as a pair $\langle I^+; I^- \rangle$ where $I^+$ and $I^-$ consist of the propositions in $\mathcal{L}$ that are mapped to *true* and to *false* respectively. (All the remaining propositions are mapped to *unknown*.)

**Definition 2.3 (3-valued interpretations).**
Let $P$ be an *edlp* written in a propositional language $\mathcal{L}$.

1. A 3-valued interpretation $I$ of $P$ is a pair $\langle I^+; I^- \rangle$ where $I^+$ and $I^-$ are disjoint subsets of $\mathcal{L}$ and such that $\mathbf{t} \in I^+$, $\mathbf{f} \in I^-$ and $\mathbf{u} \notin I^+ \cup I^-$.
2. A proposition $a \in \mathcal{L}$ is *true* in $I$ if $a \in I^+$; $a$ is *false* in $I$ if $a \in I^-$; and $a$ is *unknown* in $I$ otherwise. The truth values of more complex sentences with respect to $I$ are computed using the Kleene truth tables (in which we have abbreviated *true*, *false* and *unknown* as $t, f$ and $u$ respectively):

| $\wedge$ | $t$ | $u$ | $f$ |
|---|---|---|---|
| $t$ | $t$ | $u$ | $f$ |
| $u$ | $u$ | $u$ | $f$ |
| $f$ | $f$ | $f$ | $f$ |

| $\vee$ | $t$ | $u$ | $f$ |
|---|---|---|---|
| $t$ | $t$ | $t$ | $t$ |
| $u$ | $t$ | $u$ | $u$ |
| $f$ | $t$ | $u$ | $f$ |

| $a$ | $t$ | $u$ | $f$ |
|---|---|---|---|
| *not* $a$ | $f$ | $u$ | $t$ |

3. The truth value of a sentence $\varphi$ with respect to an interpretation $I$ is denoted by $\mathcal{V}_I(\varphi)$.
4. $I^{\mathbf{u}}$ denotes $\mathcal{L} - (I^+ \cup I^-)$, i.e., the set of propositions that are *unknown* in $I$.

Based on the orderings on truth values given before, the 3-valued interpretations can be ordered in the following ways.

**Definition 2.4 (Orderings among 3-valued interpretations).**
Let $P$ be an *edlp*. Given two 3-valued interpretations $I = \langle I^+; I^- \rangle$ and $J = \langle J^+; J^- \rangle$, the following are two possible ways of ordering $I$ and $J$:

1. *Truth Ordering* ($\preceq_t$) *on 3-valued interpretations*:
$$I \preceq_t J \text{ iff } \mathcal{V}_I(a) \leq_t \mathcal{V}_J(a) \text{ for all } a \in \mathcal{L}.$$
2. *Knowledge Ordering* ($\preceq_k$) *on 3-valued interpretations*:
$$I \preceq_k J \text{ iff } \mathcal{V}_I(a) \leq_k \mathcal{V}_J(a) \text{ for all } a \in \mathcal{L}.$$

Equivalent definitions of these orderings that appear frequently in the literature (see e.g [Prz91]) are $I \preceq_t J$ iff $I^+ \subseteq J^+$ and $I^- \supseteq J^-$; and $I \preceq_k J$ iff $I^+ \subseteq J^+$ and $I^- \subseteq J^-$.

As usual, a model of an *edlp* is an interpretation that satisfies all the clauses of the program.

**Definition 2.5 (3-valued (minimal) models).**
Let $P$ be an *edlp*.

1. A 3-valued interpretation $M$ is a 3-valued model of $P$ if for every clause $H \leftarrow B$ in $P$ , $\mathcal{V}_M(H) \geq_t \mathcal{V}_M(B)$.
2. $M$ is said to be a $\prec_t$*-minimal* (respectively $\prec_k$*-minimal*) 3-valued model of $P$ if there is no 3-valued model $N$ of $P$ such that $N \neq M$ and $N \prec_t M$ (respectively $N \prec_k M$).

A semantics of an *edlp* is captured by a subcollection of its set of models. In particular, the 3-valued stable model semantics of an *edlp* is given by the set of its 3-valued stable models as defined below.

**Definition 2.6 (3-valued (or Partial) Stable Model [Prz91]).**
Let $P$ be an *edlp* and let $M$ be any 3-valued model of $P$.

1. The *Gelfond–Lifschitz* transformation $P^M$ of $P$ with respect to $M$ is the *edlp* free of negation-by-default obtained by replacing in every clause of $P$ all negated-by-default premises $l = not\ c$ which are *true* (respectively *unknown*; respectively *false*) in $M$ by the proposition $\mathbf{t}$ (respectively $\mathbf{u}$; respectively $\mathbf{f}$).
2. $M$ is a *3-valued (or partial) stable model* of $P$ if $M$ is a $\prec_t$-minimal model of $P^M$.

Given an *edlp* $P$, Przymusinski proved the following relationships among the collections of partial stable models *3-STABLE(P)*, stable models *2-STABLE(P)* and the well-founded model *WFS(P)* of $P$.

**Proposition 2.1 ([Prz91]).**
*Let $P$ be an edlp and let $M$ be a 3-valued model of $P$.*

1. *If $M \in$ 3-STABLE(P) then $M$ is a $\prec_t$-minimal 3-valued model of $P$.*
2. *If $M \in$ 2-STABLE(P) then $M \in$ 3-STABLE(P).*
3. *If $P$ is a normal logic program and $M =$ WFS(P) then $M \in$ 3-STABLE(P). In addition, $M$ is $\prec_k$-minimal among the partial stable models of $P$, i.e. for all $N \in$ 3-STABLE(P), $M \preceq_k N$.*

Notice that the notion of partial stability is defined using the truth ordering $\prec_t$, and henceforth we consider only this ordering.

# 3 Characterization of Partial Stable Models of *edlps*

In this section we prove a new characterization of the partial stable model semantics in terms of well-supported 3-valued models of *edlps*. As stated in the introduction, the notion of well-supported 2-valued models was introduce by Fages ([Fag91]) for the class of normal logic programs. In section 3.2 we summarize the relevant definitions in [Fag91] and extend that notion to *edlps* and to the 3-valued case. We show that the notions of partial stability and 3-valued well-supportedness are equivalent. The proof of this characterization is based in the existence of a fixpoint operator that computes the $\prec_t$-minimal 3-valued models of a positive *edlp*. We introduce such an operator in section 3.1.

## 3.1 Computing Minimal Partial Models of Positive *edlps*

We define a fixpoint operator $\tilde{T}_P$ which computes the 3-valued $\prec_t$-minimal models of an *edlp* free of negation-by-default $P$. It is worth noticing that the Fitting immediate consequence operator ([Fit85]) for the 3-valued case computes the $\prec_k$-minimal models of $P$ and so a different operator is needed to compute with respect to the truth ordering $\prec_t$.

**Definition 3.1 ($\langle Dom, \prec_t \rangle$).**

1. A set of interpretations $\mathcal{I}$ is called *canonical* if all interpretations in $\mathcal{I}$ are $\prec_t$-incomparable, i.e. if for all distinct $I, J \in \mathcal{I}, I \not\prec_t J$ and $J \not\prec_t I$.
2. Consider the partially ordered set $\langle Dom, \prec_t \rangle$ defined by:
   - $Dom$ is the collection of all sets of canonical interpretations in the language $\mathcal{L}$.
   - the order $\prec_t$ on interpretations is extended to $Dom$ as follows: Given two canonical sets of interpretations $\mathcal{I}, \mathcal{J} \in Dom$,

     $$\mathcal{I} \preceq_t \mathcal{J} \text{ iff for all } J \in \mathcal{J} \text{ there exists } I \in \mathcal{I} \text{ such that } I \preceq_t J.$$

Given a set of interpretations $\mathcal{I}$ we define $min(\mathcal{I})$ as the subset of $\mathcal{I}$ containing just the $\prec_t$-minimal 3-valued interpretations in $\mathcal{I}$. Notice that $min(\mathcal{I})$ is a canonical set of interpretations.

It is straightforward to check that $\langle Dom, \prec_t \rangle$ is a lower semi-lattice whose bottom element is $\perp = \langle \{\mathbf{t}\}; \mathcal{L} - \{\mathbf{t}, \mathbf{u}\} \rangle$, whose top element is $\top = \langle \mathcal{L} - \{\mathbf{f}, \mathbf{u}\}; \{\mathbf{f}\} \rangle$ and where the greatest lower bound (glb) of a collection $X$ of canonical sets of interpretations is given by: glb($X$)= $min(\cup X)$.

**Definition 3.2 ($T_P$ operator).**
Let $P$ be an *edlp* free of negation-by-default and $C = a_1 \vee \ldots \vee a_k \leftarrow b_1, \ldots, b_m$ be a clause in $P$. Let $B$ denote the body of $C$, i.e. $B = b_1, \ldots, b_m$. Given an interpretation $I$ of $P$, we define the *operator* $T_P$ on $I$ and $C$ as the following set of interpretations:

$$T_P(I,C) = \begin{cases} min[\{\langle I^+ \cup \{a_i\}; I^- - \{a_i\}\rangle : 1 \leq i \leq k\}], & \text{if } \mathcal{V}_I(B) = \mathit{true} \\ min[\{\langle I^+; I^- - \{a_i\}\rangle : 1 \leq i \leq k\}], & \text{if } \mathcal{V}_I(B) = \mathit{unknown} \\ \{I\}, & \text{if } \mathcal{V}_I(B) = \mathit{false} \end{cases}$$

**Lemma 3.1.**
*Let $C$ an arbitrary but fixed clause. Then, $T_P(\_, C)$ is monotonic on its first argument. In other words, given interpretations $I$ and $J$, if $I \preceq_t J$ then $T_P(I,C) \preceq_t T_P(J,C)$.*

*Proof.* Let $C$ be of the form $a_1 \vee \ldots \vee a_k \leftarrow B$. Note first that for any interpretation $I$, $\{I\} \preceq_t T_P(I,C)$. Let $I$ and $J$ be interpretations such that $I \preceq_t J$. We need to show that $T_P(I,C) \preceq_t T_P(J,C)$, that is, that for every $J' \in T_P(J,C)$ there is an $I' \in T_P(I,C)$ such that $I' \preceq_t J'$.

**Case 1:** $\mathcal{V}_I(B) = \mathit{true}$. This implies that $\mathcal{V}_J(B) = \mathit{true}$ and hence, if $J' \in T_P(J,C)$, $J'$ is of the form $\langle J^+ \cup \{a_i\}; J^- - \{a_i\}\rangle$ for some $i \in \{1, \ldots, k\}$. Since $I \preceq_t J$ then $I' = \langle I^+ \cup \{a_i\}; I^- - \{a_i\}\rangle \preceq_t J'$.

**Case 2:** $\mathcal{V}_I(B) = \mathit{unknown}$. This implies that $\mathcal{V}_J(B)$ is either *true* or *unknown*. Let $J' \in T_P(J,C)$. If $\mathcal{V}_J(B)$ is *true* then $J' = \langle J^+ \cup \{a_i\}; J^- - \{a_i\}\rangle$ and if $\mathcal{V}_J(B)$ is *unknown* then $J' = \langle J^+; J^- - \{a_i\}\rangle$, for some $i \in \{1, \ldots, k\}$. In both cases, $I' = \langle I^+; I^- - \{a_i\}\rangle \preceq_t J'$.

**Case 3:** $\mathcal{V}_I(B) = \mathit{false}$. This implies that $I' = I \preceq_t J \preceq_t J'$ for all $J' \in T_P(J,C)$.

In any of these three cases, either $I' \in T_P(I,C)$ or there is some $I'' \in T_P(I,C)$ such that $I'' \preceq_t I' \preceq_t J'$. □

**Definition 3.3 ($\tilde{T}_P$ operator).**
Let $P$ be an *edlp* free of negation-by-default and let $\{C_1, \ldots, C_n\}$, for some $n \geq 0$, be the set of clauses in $P$. The *operator* $\tilde{T}_P$ *on* $\langle Dom, \prec_t\rangle$ is defined as follows: Given a canonical set of interpretations $\mathcal{I}_0$, consider the sequence of canonical sets of interpretations $\langle \mathcal{I}_0, \ldots \mathcal{I}_n\rangle$, defined inductively by:

$$\mathcal{I}_{i+1} = min[\bigcup_{I \in \mathcal{I}_i} T_P(I, C_{i+1})],$$

then $\tilde{T}_P(\mathcal{I}_0) = \mathcal{I}_n$.

**Proposition 3.1.**
*$\tilde{T}_P$ is monotonic on $\langle Dom, \prec_t\rangle$.*

*Proof.* Given $\mathcal{I}, \mathcal{J} \in Dom$, it is enough to show that $\mathcal{I} \preceq_t \mathcal{J}$ implies that $min[\bigcup_{I \in \mathcal{I}} T_P(I, C)] \preceq_t min[\bigcup_{J \in \mathcal{J}} T_P(J, C)]$ for every clause $C$ in $P$. Assume $\mathcal{I} \preceq_t \mathcal{J}$ and let $J' \in min[\bigcup_{J \in \mathcal{J}} T_P(J, C)]$. Then $J' \in T_P(J, C)$ for some $J \in \mathcal{J}$. By hypothesis, there is some $I \in \mathcal{I}$ such that $I \preceq_t J$. By Lemma 3.1, $T_P(I, C) \preceq_t T_P(J, C)$ and therefore there is some $I' \in T_P(I, C)$ such that $I' \preceq_t J'$. Since $I' \in \bigcup_{I \in \mathcal{I}} T_P(I, C)$ then there is some $I'' \in min[\bigcup_{I \in \mathcal{I}} T_P(I, C)]$ for which $I'' \preceq_t I' \preceq_t J'$. $\square$

**Lemma 3.2.**
*Let $P$ be an edlp, $M$ a 3-valued model of $P$ and $\mathcal{I}$ a canonical set of interpretations. For every clause $C \in P$, if $\mathcal{I} \preceq_t \{M\}$ then $min[\bigcup_{I \in \mathcal{I}} T_P(I, C)] \preceq_t \{M\}$.*

*Proof.* Let $C = a_1 \vee \ldots \vee a_k \leftarrow B$. Assume that $\mathcal{I} \preceq_t \{M\}$ and let $I \in \mathcal{I}$ such that $I \preceq_t M$.

**Case 1** If $\mathcal{V}_I(B) = true$ then $\mathcal{V}_M(B) = true$ and therefore there is some $a_i \in M^+$ since $M$ models $C$. Hence, $I' = \langle I^+ \cup \{a_i\}; I^- - \{a_i\} \rangle \preceq_t M$.

**Case 2** If $\mathcal{V}_I(B) = unknown$ then $\mathcal{V}_M(B)$ is either $true$ or $unknown$. Hence, there is some $a_i$ in the head of $C$ for which $\mathcal{V}_M(a_i) \succeq_t unknown$ and so $I' = \langle I^+; I^- - \{a_i\} \rangle \preceq_t M$.

**Case 3** If $\mathcal{V}_I(B) = false$ then $I' = I \preceq_t M$.

In any of these three cases $I' \in \bigcup_{I \in \mathcal{I}} T_P(I, C)$. Therefore, there is some $I'' \in min[\bigcup_{I \in \mathcal{I}} T_P(I, C)]$ such that $I'' \preceq_t I' \preceq_t M$. $\square$

Since the operator $\tilde{T}_P$ is monotonic on the lower semi-lattice $\langle Dom, \prec_t \rangle$, then it has a least fixed point on the semi-lattice. Furthermore this least fixed point is given by $\tilde{T}_P \uparrow^{\infty} (\perp)$. The following result shows that this least fixed point consists of the set of $\prec_t$-minimal 3-valued models of $P$.

**Theorem 3.3.**
*$\tilde{T}_P \uparrow^{\infty} (\perp)$ is the canonical set of $\prec_t$-minimal 3-valued models of $P$.*

*Proof.* By construction, every $I \in \tilde{T}_P \uparrow^{\infty} (\perp)$ satisfies all the clauses in $P$ and therefore $I$ is a 3-valued model of $P$. Now, let $M$ be a 3-valued model of $P$. Since $\perp \preceq_t M$, a simple induction together with Lemma 3.2 shows that $\tilde{T}_P \uparrow^{\infty} (\perp) \preceq M$. Hence, there exists a 3-valued model $M_0 \in \tilde{T}_P \uparrow^{\infty} (\perp)$ such that $M_0 \preceq M$. This implies that if $M$ is a $\prec_t$-minimal 3-valued model of $P$ then $M \in \tilde{T}_P \uparrow^{\infty} (\perp)$. In other words, $\tilde{T}_P \uparrow^{\infty} (\perp)$ contains all the $\prec_t$-minimal 3-valued models of $P$. Since $\tilde{T}_P \uparrow^{\infty} (\perp)$ is a canonical set of interpretations, $\tilde{T}_P \uparrow^{\infty} (\perp)$ cannot contain any other model of $P$. Hence $\tilde{T}_P \uparrow^{\infty} (\perp)$ contains precisely the $\prec_t$-minimal 3-valued models of $P$. $\square$

**Example 3.1.** *Consider the following positive edlp $P$.*

$$
\begin{array}{llrcll}
P = \{ & C_1: & a \vee b & \leftarrow & c & \\
& C_2: & d & \leftarrow & & \\
& C_3: & c & \leftarrow & d, \mathbf{u} & \}
\end{array}
$$

$\tilde{T}_P(\bot)$ *:*

$T_P(\bot, C_1) = \{I_1 = \bot\} = \mathcal{I}_1$

$T_P(I_1, C_2) = \{I_2 = \langle\{\mathbf{t}, d\}; \{\mathbf{f}, a, b, c\}\rangle\} = \mathcal{I}_2$

$T_P(I_2, C_3) = \{I_3 = \langle\{\mathbf{t}, d\}; \{\mathbf{f}, a, b\}\rangle\} = \mathcal{I}_3$

*So,* $\tilde{T}_P(\bot) = \mathcal{I}_3$.

$\tilde{T}_P\uparrow^2(\bot)$ *:*

$T_P(I_3, C_1) = \{I_4 = \langle\{\mathbf{t}, d\}; \{\mathbf{f}, a\}\rangle, I_5 = \langle\{\mathbf{t}, d\}; \{\mathbf{f}, b\}\rangle\} = \mathcal{I}_4$

$T_P(I_4, C_2) = \{I_4\}$ $\qquad$ $T_P(I_5, C_2) = \{I_5\}$

$T_P(I_4, C_3) = \{I_4\}$ $\qquad$ $T_P(I_5, C_3) = \{I_5\}$

*So,* $\tilde{T}_P\uparrow^2(\bot) = \mathcal{I}_4$.

$\tilde{T}_P\uparrow^3(\bot) = \tilde{T}_P\uparrow^2(\bot) = \mathcal{I}_4$.

*Hence $P$ has two $\prec_t$-minimal 3-valued models, namely $I_4$ and $I_5$.*

As shown in Theorem 3.3, $\tilde{T}_P\uparrow^\infty(\bot)$ is the set of $\prec_t$-minimal 3-valued models of $P$. Therefore the least fixed point of $\tilde{T}_P$ is independent of the ordering of the clauses in the program.

Finally, we point out that for a positive *edlp* $P$ in which the proposition $\mathbf{u}$ does not appear, $\tilde{T}_P\uparrow^\infty(\bot)$ consists of the set of minimal (2-valued) models of $P$, and so, for this case the least fixed point of $\tilde{T}_P$ coincides with the least fixed point of the Minker/Rajasekar fixpoint operator ([MR90]).

## 3.2 Well-Supported 3-valued Models

We start this section by briefly surveying the definition of 2-valued well-supported models given by Fages [Fag91] and his characterization of the 2-valued stable model semantics. Then we introduce our extended definition and characterization for the disjunctive 3-valued case.

**Definition 3.4 (Well-supported 2-valued interpretations [Fag91]).**
A Herbrand interpretation $I$ is a *well-supported* 2-valued interpretation of a normal logic program $P$ iff there exists a strict well-founded partial ordering $<$ on $I$ such that for any $a \in I$ there is a ground instance of a clause

$$C = a \leftarrow \underbrace{b_1, \ldots, b_m, not\ c_1, \ldots, not\ c_n}_{B}$$

in $P$ satisfying the following conditions:

1. $a > b_i$ for all $i \in \{1, \dots, m\}$ and
2. $\mathcal{V}_I(B) = true$.

**Theorem 3.4 (([Fag91])).**
*Let $P$ be a normal logic program and let $M$ be a Herbrand interpretation of $P$. Then, $M$ is a stable model of $P$ iff $M$ is a well-supported model of $P$.*

Condition 2 guarantees that $C$ is a support for $a$ to be *true*. Condition 1 guarantees that this support is loop-free, that is, the justifications for the $b$'s to be *true* do not depend on the fact that $a$ is *true*. We extend those conditions to disjunctive clauses.

**Definition 3.5 (Well-supported 3-valued interpretations).**
A Herbrand 3-valued interpretation $I$ is a *well-supported 3-valued interpretation* of an *edlp* $P$ iff there exists a strict well-founded partial ordering $<$ on $I^+ \cup I^{\mathbf{u}}$ such that for any $a \in I^+ \cup I^{\mathbf{u}}$ there is a ground instance of a clause

$$C = a \vee \underbrace{a_1 \vee \ldots \vee a_k}_{H} \leftarrow \underbrace{b_1, \dots, b_m, not\ c_1, \dots, not\ c_n}_{B}$$

in $P$ satisfying the following conditions:

1. $a > b_i$ for all $i \in \{1, \dots, m\}$ and
2. **Case 1:** If $a \in I^+$, then $\mathcal{V}_I(B) = true$ and $\mathcal{V}_I(H) <_t true$, or
   **Case 2:** If $a \in I^{\mathbf{u}}$, then $\mathcal{V}_I(B) = unknown$ and $\mathcal{V}_I(H) = false$.
   (These two cases can be summarized as: $\mathcal{V}_I(H) <_t \mathcal{V}_I(B) = \mathcal{V}_I(a)$.)

The 3-valued well-supported models of an *edlp* $P$ are exactly the 3-valued stable models of the program as the following theorem shows. The proof of this result is based on the fact that a 3-valued stable model $M$ of $P$ is a $\prec_t$-minimal 3-valued model of $P^M$ and therefore it can be constructed using the fixpoint operator $\tilde{T}_{P^M}$ defined in section 3.1 whose iterations provide a well-founded order on $M^+ \cup M^{\mathbf{u}}$.

**Theorem 3.5.**
*Let $P$ be an edlpand let $M$ be a 3-valued interpretation of $P$. Then, $M$ is a 3-valued stable model of $P$ iff $M$ is a well-supported 3-valued model of $P$.*

*Proof.* "$\Rightarrow$" Let $C_0, \dots, C_{n-1}$ list all the clauses in $P$. If $M$ is a 3-valued stable model of $P$ then $M$ is a $\prec_t$-minimal 3-valued model of $P^M$. Then $M$ can be rebuilt using the fixpoint operator $\tilde{T}_{P^M}$. Let $\alpha$ be the smallest ordinal for which $\tilde{T}_{P^M} \uparrow^{\alpha} (\bot)$ is the least fixed point of $\tilde{T}_{P^M}$. Let

$$\langle \bot = M_0, \underbrace{M_1, \dots, M_n}_{\tilde{T}_{P^M}\uparrow^1(\bot)}, \underbrace{M_{n+1}, \dots, M_{2n}}_{\tilde{T}_{P^M}\uparrow^2(\bot)}, \dots, \underbrace{M_{(\alpha-1)n+1}, \dots, M_{\alpha n}}_{\tilde{T}_{P^M}\uparrow^{\alpha}(\bot)} = M \rangle$$

be a trace of the construction of $M$, i.e. a sequence of interpretations that converges to $M$ and such that for all $i$, $M_i \preceq_t M$ and $M_{i+1} \in T_P(M_i, C_{i \bmod n})$, where mod$n$ denotes the modulo $n$ function. Such a trace exists due to Lemma 3.2. Given an element $a \in M^+ \cup M^{\mathbf{u}}$, we say that the *rank* of $a$, denoted by rank$(a)$, with respect to the trace is $i$ if $i$ is the smallest integer for which $\mathcal{V}_{M_i}(a) = \mathcal{V}_M(a)$ (notice that the rank of every element in $M^+ \cup M^{\mathbf{u}}$ is always greater than 0). Let $<$ be the strict well-founded partial ordering on $M^+ \cup M^{\mathbf{u}}$ given by

$$a < b \text{ iff } \begin{cases} a, b \in M^+ \text{ and } \mathrm{rank}(a) < \mathrm{rank}(b) & \text{or} \\ a, b \in M^{\mathbf{u}} \text{ and } \mathrm{rank}(a) < \mathrm{rank}(b) & \text{or} \\ a \in M^+ \text{ and } b \in M^{\mathbf{u}}. \end{cases}$$

This order is a well-supported order on $M$. To see this, let $a \in M^+ \cup M^{\mathbf{u}}$ and suppose that $a$ is of rank $i+1$. By definition of $\tilde{T}_{P^M}$, there is a clause

$$C = a \vee \underbrace{a_1 \vee \ldots \vee a_k}_{H} \leftarrow \underbrace{b_1, \ldots, b_m, \mathit{not}\ c_1, \ldots, \mathit{not}\ c_n}_{B}$$

in $P$ and consequently there is a clause

$$C^M = a \vee \underbrace{a_1 \vee \ldots \vee a_k}_{H} \leftarrow \underbrace{b_1, \ldots, b_m, \mathcal{V}_M(\mathit{not}\ c_1), \ldots, \mathcal{V}_M(\mathit{not}\ c_n)}_{B^M}$$

in $P^M$ such that:

**Case 1:** If $a \in M^+_{i+1}$, then $\mathcal{V}_{M_i}(B^M) = \mathit{true}$, and so $\mathcal{V}_M(B) = \mathit{true}$, which implies that $a > b_i$ for all $i \in \{1, \ldots, m\}$. Furthermore $C$ can be selected in such a way that $\mathcal{V}_{M_i}(H) <_t \mathit{true}$, since otherwise $\langle M^+ - \{a\}; M^- \rangle$ would be a model of $P^M$ contradicting the $\prec_t$-minimality of $M$.

**Case 2:** If $a \in M^{\mathbf{u}}_{i+1}$, then $\mathcal{V}_{M_i}(B^M) = \mathit{unknown}$ which implies that $a > b_i$ for all $i \in \{1, \ldots, m\}$. Furthermore $C$ can be selected in such a way that $\mathcal{V}_{M_i}(H) = \mathit{false}$, because otherwise $\langle M^+; M^- \cup \{a\} \rangle$ would be a model of $P^M$ contradicting the $\prec_t$-minimality of $M$.

Hence, $M$ is a well-supported 3-valued model of $P$.

"$\Leftarrow$" Let $M$ be a well-supported model of $P$. Since $M$ is a model of $P$ then $M$ is a model of $P^M$. We need to show that $M$ is a $\prec_t$-minimal model of $P^M$. Assume that $M$ is not a $\prec_t$-minimal model of $P^M$. Let $N$ be a $\prec_t$-minimal model of $P^M$ such that $N \prec_t M$. Let $a$ be a smallest element (with respect to the well-founded order $<$) for which $\mathcal{V}_N(a) \prec_t \mathcal{V}_M(a)$.

Since $M$ is well-supported, there is a clause

$C = a \vee \underbrace{a_1 \vee \ldots \vee a_k}_{H} \leftarrow \underbrace{b_1, \ldots, b_m, \mathit{not}\ c_1, \ldots, \mathit{not}\ c_n}_{B}$ in $P$ such that:

1. $a > b_i$ for all $i \in \{1, \ldots, m\}$ and
2. **Case 1:** If $a \in M^+$, then $\mathcal{V}_M(B) = \mathit{true}$ and $\mathcal{V}_M(H) <_t \mathit{true}$. Since $a > b_i$ for all $i \in \{1, \ldots, m\}$, then the truth-values of the $b$'s are the same under $M$ and under $N$ so $\mathcal{V}_N(B^M) = \mathcal{V}_M(B) = \mathit{true}$ and since $N \prec_t M$ then $\mathcal{V}_N(H) \leq_t \mathcal{V}_M(H) <_t \mathit{true}$ and so if $\mathcal{V}_N(a) <_t \mathcal{V}_M(a)$, $N$ would not be a model of $P^M$ contradicting the choice of $N$.

   **Case 2:** If $a \in M^{\mathbf{u}}$, then $\mathcal{V}_M(B) = \mathit{unknown}$ and $\mathcal{V}_M(H) = \mathit{false}$. Since $a > b_i$ for all $i \in \{1, \ldots, m\}$, then $\mathcal{V}_N(B^M) = \mathcal{V}_M(B) = \mathit{unknown}$ and since $N \prec_t M$ then $\mathcal{V}_N(H) = \mathit{false}$. Therefore, if $\mathcal{V}_N(a) <_t \mathit{unknown}$ then $N$ is not a model of $P^M$ which is a contradiction.

Hence, $M$ is a $\prec_t$-minimal model of $P^M$.

□

# 4 Computing Partial Stable Models of *edlps*

This section is concerned with the computation of the 3-valued stable models of an *edlp*. We introduce a transformation, called the *3S–transformation*, that, given an *edlp* $P$, computes a new *edlp* $P^{3S}$ free of negation-by-default whose set of minimal 2-valued models corresponds to the 3-valued stable models of the original program. An algorithm to compute the minimal 2-valued models of an *edlp* free of negation-by-default is given in section 4.2.

## 4.1 The 3S–transformation

Given an *edlp* $P$, the 3S–transformation performs case analysis to construct all potential justifications or supports for a proposition to be *true*, *false* or *unknown*. Those justifications are written as constrained clauses and collected to form a positive *edlp* called $P^{3S}$. The constraints ensure that the justifications are loop-free.

$P^{3S}$ is written in a richer language $\hat{\mathcal{L}}$ which is obtained by adding to $\mathcal{L}$ new propositional symbols $ua$ and $na$ for each propositional symbol $a \in \mathcal{L}$. Intuitively, $a$ will be understood as $a$ is *true*, $ua$ as $a$ is *unknown* and $na$ as $a$ is *false*.

**Definition 4.1 (Extended language $\hat{\mathcal{L}}$).**
Let $\mathcal{L}$ be a propositional language. $\mathcal{L}$ is extended to the propositional language $\hat{\mathcal{L}} = \{a, ua, na | a \in \mathcal{L}\}$.

We introduce operators $\mathcal{T}, \mathcal{F}$ and $\mathcal{U}$ which, applied to a sentence in the language $\mathcal{L}$, produce sets of all possible justifications in the expanded language $\hat{\mathcal{L}}$ under which the given sentence is *true*, *false* or *unknown* respectively. In other words, a sentence $\varphi$ is *true* (resp. *false*, resp. *unknown*) if and only if at least one of the supporting sentences in $\mathcal{T}(\varphi)$ (resp. $\mathcal{F}(\varphi)$, resp. $\mathcal{U}(\varphi)$) holds.

In what follows we inductively define these operators.

**Definition 4.2 (Operators $\mathcal{T},\mathcal{F}$ and $\mathcal{U}$ on normal literals).**
Let $a \in \mathcal{L}$. The *operators* $\mathcal{T},\mathcal{F}$ *and* $\mathcal{U}$ are defined on $a$ and on *not* $a$ as follows:

$$\begin{array}{ll} \mathcal{T}(a) = \{a\} & \mathcal{T}(not\ a) = \{na\} \\ \mathcal{U}(a) = \{ua\} & \mathcal{U}(not\ a) = \{ua\} \\ \mathcal{F}(a) = \{na\} & \mathcal{F}(not\ a) = \{a\} \end{array}$$

A disjunction of propositions $H = a_1 \vee \ldots \vee a_k$, is *true* when at least one of the propositions $a_1, \ldots, a_k$ is *true*; *false* when all these propositions are *false* and *unknown* when at least one of these propositions is *unknown* and the remaining ones are either *unknown* or *false*. We codify all possibilities under which $H$ is unknown by using $k$-tuples of 0's and 1's that contain at least one 1. Such a tuple $\langle \lambda_1, \ldots, \lambda_k \rangle$ can be seen as stating that $a_i$ is *false* if $\lambda_i = 0$ and *unknown* if $\lambda_i = 1$. If at least one $\lambda_j$ is 1, then $H$ is *unknown*. We express this formally in the language $\hat{\mathcal{L}}$ in the following definition.

**Definition 4.3 (Operators $\mathcal{T},\mathcal{F}$ and $\mathcal{U}$ on disjunctions).**
Let $H = a_1 \vee \ldots \vee a_k$, $k \geq 0$, be an arbitrary disjunction of propositions. The *operators* $\mathcal{T},\mathcal{F}$ *and* $\mathcal{U}$ are defined on $H$ as follows:

$\mathcal{T}(H) = \{a_1 \mid \ldots \mid a_k\}$ [2]
$\mathcal{F}(H) = \{na_1 \wedge \ldots \wedge na_k\}$
$\mathcal{U}(H) = \{(\mathcal{F}/\mathcal{U})^{\lambda_1}(a_1) \wedge \ldots \wedge (\mathcal{F}/\mathcal{U})^{\lambda_k}(a_k) : \langle \lambda_1, \ldots, \lambda_k \rangle \in \mathcal{B}^k\}$

where:

- $\mathcal{B}^k = \{\langle \lambda_1, \ldots, \lambda_k \rangle : \lambda_1, \ldots, \lambda_k \in \{0,1\} \text{ and } \exists j \in \{1, \ldots, k\}, \lambda_j = 1\}$.
- $(\mathcal{F}/\mathcal{U})^{\lambda}(a) = \begin{cases} \mathcal{F}(a), & \text{If } \lambda = 0 \\ \mathcal{U}(a), & \text{If } \lambda = 1 \end{cases}$

Notice that when $H$ is an empty disjunction (i.e. when $k = 0$) the previous definition makes $\mathcal{T}(H) = \mathcal{U}(H) = \{\} \equiv \{\mathbf{f}\}$ and $\mathcal{F}(H) = \{\mathbf{t}\}$.

We follow a similar process to define the truth value of a conjunction of normal literals $B = b_1, \ldots, b_m, not\ c_1, \ldots, not\ c_n$. $B$ is *true* if all $b$'s are *true* and all $c$'s are *false*. It is *false* if at least one of the $b$'s is *false* or one of the $c$'s is *true*. And it is *unknown* if the truth values of the $b$'s and (*not* $c$)'s are greater than or equal to *unknown* (i.e. *unknown* or *true*) and at least one of them is *unknown*. Again, we codify all possibilities under which $B$ is *unknown* by using $(m+n)$-tuples of 0's and 1's that contain at least one 1. Such a tuple $\langle \lambda_1, \ldots, \lambda_{m+n} \rangle$ can be seen as stating that the $b$'s and the (*not* $c$)'s are *true* if the corresponding entries in the tuple equal 0 or are *unknown* if they are equal

[2] We use the symbol "|" to separate elements in a set.

to 1. Since at least one entry is 1, then $B$ is *unknown*. The following definition formalizes this in the language $\hat{\mathcal{L}}$.

**Definition 4.4 (Operators $\mathcal{T},\mathcal{F}$ and $\mathcal{U}$ on conjunctions).**
Let $B = b_1, \ldots, b_m, not\ c_1, \ldots, not\ c_n$, where $m, n \geq 0$. The *operators* $\mathcal{T},\mathcal{F}$ *and* $\mathcal{U}$ are defined on $B$ as follows:

$$\mathcal{T}(B) = \{b_1 \wedge \ldots \wedge b_m \wedge nc_1 \wedge \ldots \wedge nc_n\}$$
$$\mathcal{F}(B) = \{nb_1 \mid \ldots \mid nb_m \mid c_1 \mid \ldots \mid c_n\}$$
$$\begin{aligned}\mathcal{U}(B) = \{\ & (\mathcal{T}/\mathcal{U})^{\lambda_1}(b_1) \wedge \ldots \wedge (\mathcal{T}/\mathcal{U})^{\lambda_m}(b_m) \wedge \\ & (\mathcal{T}/\mathcal{U})^{\lambda_{m+1}}(not\ c_1) \wedge \ldots \wedge (\mathcal{T}/\mathcal{U})^{\lambda_{m+n}}(not\ c_n) : \\ & \langle \lambda_1, \ldots, \lambda_{m+n} \rangle \in \mathcal{B}^{m+n} \}\end{aligned}$$

where:

$$(\mathcal{T}/\mathcal{U})^{\lambda}(\varphi) = \begin{cases} \mathcal{T}(\varphi), & \text{If } \lambda = 0 \\ \mathcal{U}(\varphi), & \text{If } \lambda = 1 \end{cases}$$

When $B$ is an empty conjunction (i.e. when $m, n = 0$), $\mathcal{T}(B) = \{\mathbf{t}\}$ and $\mathcal{F}(B) = \mathcal{U}(B) = \{\} \equiv \{\mathbf{f}\}$, according to the previous definition.

We concentrate now on determining when a clause is a support for a proposition with respect to a model $M$ of the clause. Assume that there is a well-founded partial order $<$ on $M^{+} \cup M^{\mathbf{u}}$. Let $a$ be an arbitrary but fixed proposition and let $C = a \vee H \leftarrow B$, where $B = b_1, \ldots, b_m, not\ c_1, \ldots, not\ c_n$. $C$ is a *support* for $a$ with respect to $M$ if one of the following cases holds.

1. If $\mathcal{V}_M(a)$ is *true* then $\mathcal{V}_M(H) <_t \mathcal{V}_M(B) = true$ and $a > b_i$ for all $i \in \{1, \ldots, m\}$.
2. If $\mathcal{V}_M(a)$ is *unknown* then $\mathcal{V}_M(H) <_t \mathcal{V}_M(B) = unknown$ and $a > b_i$ for all $i \in \{1, \ldots, m\}$.
3. If $\mathcal{V}_M(a)$ is *false* then $\mathcal{V}_M(H) \geq_t \mathcal{V}_M(B)$ (this happens when $\mathcal{V}_M(H)$ is *true*, when $\mathcal{V}_M(B)$ is *false* or when both values are *unknown*).

These three cases are explicitly coded in the operators $\tilde{\mathcal{T}}_a, \tilde{\mathcal{U}}_a$ and $\tilde{\mathcal{F}}_a$ in the following definition. A set of constraints $\{a > b_i : 1 \leq i \leq m\}$ with respect to a clause $C = a \vee H \leftarrow b_1, \ldots, b_m, not\ c_1, \ldots, not\ c_n$ can be understood as requiring that if the clause $C$ is used to support that $a$ is either *true* or *unknown*, then the proofs that the $b$'s are *true* or *unknown* should not rely on the proof for $a$. Then we say that a set of constraints is satisfied when $<$ is a partial order (i.e. $a > b$ and $b > a$ are not required simultaneously). Since the definition of well-supportedness calls only for the existence of a partial order in the set of *true* and *unknown* propositions of a model, we do not have to add constraints to clauses supporting $a$ to be *false*.

**Definition 4.5 (Operators $\tilde{\mathcal{T}}_a, \tilde{\mathcal{F}}_a$, and $\tilde{\mathcal{U}}_a$).**
Let $a \in \mathcal{L}$ and let $C = a \vee H \leftarrow \underbrace{b_1, \ldots, b_m, not\ c_1, \ldots, not\ c_n}_{B}$ be a clause in $P$.

- Let $\mathcal{C}_a(B)$ be the following set of constraints:

$$\mathcal{C}_a(B) = \{a > b_i : 1 \leq i \leq m\}.$$

- The *operators* $\tilde{\mathcal{T}}_a, \tilde{\mathcal{F}}_a$, *and* $\tilde{\mathcal{U}}_a$ on the clause $C$ are defined as follows:

  $\tilde{\mathcal{T}}_a(H,B) = \mathcal{T}(B), [\mathcal{F}(H) \mid \mathcal{U}(H)]$ under constraints $\mathcal{C}_a(B)$.
  $\tilde{\mathcal{F}}_a(H,B) = \mathcal{F}(B) \mid \mathcal{T}(H) \mid (\mathcal{U}(B), \mathcal{U}(H))$ under no constraints.
  $\tilde{\mathcal{U}}_a(H,B) = (\mathcal{U}(B), \mathcal{F}(H))$ under constraints $\mathcal{C}_a(B)$.

  The operators "," and "|" between sets stand for the usual operators "x" (Cartesian product) and "∪" (union) respectively. (We use here "," and "|" to preserve the flavor of logic programming syntax.)

**Example 4.1.** *Let* $C = a \leftarrow b, not\ c$. *Then, all possible supports for the three possible truth values of* $a$ *are listed below:*
$\tilde{\mathcal{T}}_a((), (b, not\ c)) = \{b \wedge nc\}$ *under* $\mathcal{C}_a(B) = \{a > b\}$,
$\tilde{\mathcal{F}}_a((), (b, not\ c)) = \{nb \mid c\}$ *under no constraints,*
$\tilde{\mathcal{U}}_a((), (b, not\ c)) = \{(ub \wedge nc) \mid (b \wedge uc) \mid (ub \wedge uc)\}$ *under* $\mathcal{C}_a(B)$,
*which state that the only justification for* $a$ *to be true is that* $b$ *be true and* $c$ *be false simultaneously. There are two supports for* $a$ *to be false, namely* $b$ *is false or* $c$ *is true. All the remaining possibilities support* $a$ *to be unknown.*

We apply now case analysis to construct all possible justifications of a proposition $a$ with respect to a program $P$. Consider the set of all clauses defining $a$ in $P$ (i.e. the set of clauses containing $a$ in their heads). With respect to a well-supported 3-valued model of $P$, $a$ is *true* when at least one of these clauses supports $a$ to be *true*, $a$ is *false* if all clauses in its definition support $a$ to be *false*, and $a$ is *unknown* when none of these clauses supports $a$ to be *true* but at least one of them supports $a$ to be *unknown*. Since one of these cases must hold, the clause $a \vee ua \vee na$ must be satisfied in the well-supported model.

It is worth noticing that if a proposition $a$ is not defined in $P$ (it does not appear in the head of any clause in $P$) then there is no support for it to be *true* or *unknown* and therefore it is taken to be *false*.

**Definition 4.6 (3S–transformation).**
Let $P$ be an *edlp*.

1. Let $a \in \mathcal{L}$. Let the definition of $a$ in $P$ consist of the following set of clauses:

$$\begin{array}{rcl} a \vee H_1 & \leftarrow & B_1 \\ & \vdots & \\ a \vee H_r & \leftarrow & B_r \end{array}$$

   where $r \geq 0$. The *3S–transformation of the definition of* $a$, denoted by $a^{3S}$, is given by the following set of clauses:

If $r = 0$ :
$$na \leftarrow$$

If $r > 0$ :
$$a \leftarrow \{\tilde{\mathcal{T}}_a(H_1, B_1) \mid \ldots \mid \tilde{\mathcal{T}}_a(H_r, B_r)\}$$
$$ua \leftarrow \{(\tilde{\mathcal{F}}_a/\tilde{\mathcal{U}}_a)^{\lambda_1}(H_1, B_1), \ldots, (\tilde{\mathcal{F}}_a/\tilde{\mathcal{U}}_a)^{\lambda_r}(H_r, B_r) : \langle\lambda_1, \ldots, \lambda_r\rangle \in \mathcal{B}^r\}$$
$$na \leftarrow \tilde{\mathcal{F}}_a(H_1, B_1), \ldots, \tilde{\mathcal{F}}_a(H_r, B_r)$$
$$a \vee ua \vee na \leftarrow$$

($\varphi \leftarrow \{\psi_1 \mid \ldots \mid \psi_n\}$ is a shorthand for the set of clauses $\left\{\begin{array}{ccc} \varphi & \leftarrow & \psi_1 \\ & \vdots & \\ \varphi & \leftarrow & \psi_n \end{array}\right\}$)

where:
$$(\tilde{\mathcal{F}}_a/\tilde{\mathcal{U}}_a)^{\lambda}(H, B) = \begin{cases} \tilde{\mathcal{F}}_a(H, B), & \text{If } \lambda = 0 \\ \tilde{\mathcal{U}}_a(H, B), & \text{If } \lambda = 1 \end{cases}$$

2. The *3S–transformation* $P^{3S}$ *of* $P$ is obtained by applying the 3S–transformation to each proposition in the language of $P$.

The number of clauses in $P^{3S}$ is, in general, exponential on the number of clauses in $P$ since all possible supports for each truth value of a proposition in $\mathcal{L}$ are considered.

As noted before, the 3S–transformation requires that each proposition $a$ assumes a truth value. However, it may be the case that, say, $a$ and $ua$ are both *true* in $P^{3S}$. Since this is clearly undesirable, we impose a set of denial rules on the models of $P^{3S}$ to eliminate such possibilities.

**Definition 4.7 (Denial rules $IC_P$).**
Let $P$ be a disjunctive logic program and let $IC_P$ denote the following set of denial rules:

$$IC_P = \{\Leftarrow a, ua; \Leftarrow a, na; \Leftarrow ua, na : a \in \mathcal{L} - \{\mathbf{t}, \mathbf{u}, \mathbf{f}\}\}.$$

An interpretation $I$ of $P^{3S}$ is any subset of $\hat{\mathcal{L}}$ satisfying the denial rules $IC_P$. $I^+, I^-$ and $I^{\mathbf{u}}$ denote respectively the positive, the negative and the uncertain parts of $I$, i.e., $I^+ = \{a \in \mathcal{L} : a \in I\}$, $I^- = \{a \in \mathcal{L} : na \in I\}$ and $I^{\mathbf{u}} = \{a \in \mathcal{L} : ua \in I\}$. $I^3$ denote the 3-valued interpretation $\langle I^+; I^-\rangle$.

Associated with each $a \in I^+ \cup I^{\mathbf{u}}$ there is a collection $\mathcal{C}_a^I$ that contains all the sets of constraints that appear in clauses supporting $a$ (or $ua$) with respect to $I$ (for an illustration see Example 4.3 below), i.e.

$$\mathcal{C}_a^I = \{\mathcal{C}_a(B) : \text{there is a clause } a \leftarrow B \text{ (or } ua \leftarrow B\text{) under constraints } \mathcal{C}_a(B) \text{ in } P^{3S} \text{ such that } \mathcal{V}_I(B) = \mathit{true}\}.$$

Let $\mathcal{C}^I = \{\mathcal{C}_a^I : a \in I^+ \cup I^{\mathbf{u}}\}$. We say that $I$ *satisfies the constraints in* $\mathcal{C}^I$ if and only if for every $a \in I^+ \cup I^{\mathbf{u}}$ there is some $\mathcal{C}_a(B_a) \in \mathcal{C}_a^I$ such that

$[\cup_{(a \in I^+ \cup I^{\mathbf{u}})} \mathcal{C}_a(B_a)]$ defines a partial order in $I^+ \cup I^{\mathbf{u}}$.

We make precise now the notion of (minimal) 2-valued models of $P^{3S}$.

**Definition 4.8 (2-valued models of $P^{3S}$).**

1. A *2-valued model of* $P^{3S}$ is a subset $M$ of $\hat{\mathcal{L}}$ satisfying all clauses in the program and the constraints in $\mathcal{C}^M$.
2. Let $M$ and $N$ be 2-valued models of $P^{3S}$. We say that $M \leq N$ iff $M^+ \subseteq N^+$ and $N^- \subseteq M^-$.

A 3-valued interpretation $J$ of $P$ can be transformed into a 2-valued interpretation $J^2$ of $P^{3S}$ by defining $J^2 = J^+ \cup \{ua : a \in J^{\mathbf{u}}\} \cup \{na : a \in J^-\}$.

The set of minimal 2-valued models of $P^{3S}$ (denoted by $\mathcal{M}_{P^{3S}}^{IC_P}$) is closely related to the set of 3-valued stable models of $P$, as the following examples show.

**Example 4.2.** *Let* $P = \{b \vee c; a \leftarrow not\ b; a \leftarrow not\ c\}$.

$$
\begin{array}{lrcll}
P^{3S} = \{ & b & \leftarrow & (uc \mid nc) & \mathcal{C}_b = \emptyset \\
& nb & \leftarrow & c & \\
& c & \leftarrow & (ub \mid nb) & \mathcal{C}_c = \emptyset \\
& nc & \leftarrow & b & \\
& a & \leftarrow & nb \mid nc & \mathcal{C}_a = \emptyset \\
& na & \leftarrow & b, c & \\
& ua & \leftarrow & (ub, c) \mid (b, uc) \mid (ub, uc) & \mathcal{C}_a = \emptyset \\
& a \vee ua \vee na & \leftarrow & & \\
& b \vee ub \vee nb & \leftarrow & & \\
& c \vee uc \vee nc & \leftarrow & & \}
\end{array}
$$

$IC_P = \{\Leftarrow x, ux; \Leftarrow x, nx; \Leftarrow ux, nx : x \in \{a, b, c\}\}$

*The minimal 2-valued models of* $P^{3S}$ *are*

$$\mathcal{M}_{P^{3S}}^{IC_P} = \{M_1 = \{a, b, nc\}, M_2 = \{a, nb, c\}\}.$$

*Here,* $\mathcal{C}^{M_1} = \{\mathcal{C}_a^{M_1} = \{\emptyset\}, \mathcal{C}_b^{M_1} = \{\emptyset\}\}$ *and* $\mathcal{C}^{M_2} = \{\mathcal{C}_a^{M_2} = \{\emptyset\}, \mathcal{C}_c^{M_2} = \{\emptyset\}\}$. *Clearly,* $M_1$ *and* $M_2$ *respectively satisfy the constraints in* $\mathcal{C}^{M_1}$ *and* $\mathcal{C}^{M_2}$ *since an empty set of constraints defines a partial order on any set.* $M_1$ *and* $M_2$ *correspond to the partial stable models of* $P$*:* 3-STABLE*(P)* $= \{\langle\{a, b\}; \{c\}\rangle, \langle\{a, c\}; \{b\}\rangle\}$.

**Example 4.3.** *Let* $P = \{a \leftarrow b; b \leftarrow a; c \leftarrow not\ a\}$.

$$
\begin{array}{llcll}
P^{3S} = \{ & a & \leftarrow & b & \mathcal{C}_a = \{a > b\} \\
& ua & \leftarrow & ub & \mathcal{C}_a = \{a > b\} \\
& na & \leftarrow & nb & \\
& b & \leftarrow & a & \mathcal{C}_b = \{b > a\} \\
& ub & \leftarrow & ua & \mathcal{C}_b = \{b > a\} \\
& nb & \leftarrow & na & \\
& c & \leftarrow & na & \mathcal{C}_c = \emptyset \\
& uc & \leftarrow & ua & \mathcal{C}_c = \emptyset \\
& nc & \leftarrow & a & \\
& a \vee ua \vee na & \leftarrow & & \\
& b \vee ub \vee nb & \leftarrow & & \\
& c \vee uc \vee nc & \leftarrow & & \}
\end{array}
$$

$IC_P = \{\Leftarrow x, ux; \Leftarrow x, nx; \Leftarrow ux, nx : x \in \{a, b, c\}\}$

*There are three minimal models of* $P^{3S}$*:*

$M_1 = \{a, b, nc\}$ *with* $\mathcal{C}^{M_1} = \{\mathcal{C}_a = \{a > b\}, \mathcal{C}_b = \{b > a\}\}$,
$M_2 = \{ua, ub, uc\}$ *with* $\mathcal{C}^{M_2} = \{\mathcal{C}_a = \{a > b\}, \mathcal{C}_b = \{b > a\}, \mathcal{C}_c = \{\emptyset\}\}$
$M_3 = \{na, nb, c\}$ *with* $\mathcal{C}^{M_3} = \{\mathcal{C}_c = \{\emptyset\}\}$.
*Notice, however that the sets of constraints on* $M_1$ *and on* $M_2$ *are unsatisfiable since* $\{a > b, b > a\}$ *is not a partial order. Therefore,* $\mathcal{M}^{IC_P}_{P^{3S}} = \{\{na, nb, c\}\}$ *which corresponds to the unique 3-valued stable (and hence well-founded) model of* $P$*, namely* $\{\langle\{c\}; \{a, b\}\rangle\}$.

Indeed, there is a one-to-one correspondence between the minimal models of the constrained logic program $P^{3S}$ and the 3-valued well-supported (and hence partial stable) models $P$ as the following theorem shows.

**Theorem 4.1.**
*Let* $P$ *be an edlp and let* $M$ *be a 3-valued interpretation of* $P$*. Then* $M$ *is a 3-valued well-supported model of* $P$ *iff* $M^2 \in \mathcal{M}^{IC_P}_{P^{3S}}$.

*Proof.* **"$\Leftarrow$"** Assume $M^2 \in \mathcal{M}^{IC_P}_{P^{3S}}$. Let $a \in \mathcal{L}$.

**Case 1:** If $a \in M^2$, then there is some clause

$$C = a \vee \underbrace{a_1 \vee \ldots \vee a_k}_{H} \leftarrow \underbrace{b_1, \ldots, b_m, not\ c_1, \ldots, not\ c_n}_{B}$$

in $P$ for which $\tilde{\mathcal{T}}_a(H, B) = true$, i.e. $\mathcal{V}_M(B) = true$, $\mathcal{V}_M(H) <_t true$ and $M^2$ satisfies the constraints $\{a > b_i : 1 \leq i \leq m\}$. If there were no such a clause, then for every clause $a \vee H \leftarrow B \in P$ either $\tilde{\mathcal{F}}_a(H, B)$ or $\tilde{\mathcal{U}}_a(H, B)$ would be $true$. Hence, either $ua$ or $na$ would belong to $M^2$ contradicting the assumption that $M^2$ satisfies $IC_P$.

**Case 2:** If $ua \in M^2$, then there exists some $\langle \lambda_1, \ldots, \lambda_r\rangle \in \mathcal{B}^r$, where $r$ is the number of clauses in the definition of $a$ in $P$, for which the conjunction $(\tilde{\mathcal{F}}_a/\tilde{\mathcal{U}}_a)^{\lambda_1}(H_1, B_1), \ldots, (\tilde{\mathcal{F}}_a/\tilde{\mathcal{U}}_a)^{\lambda_r}(H_r, B_r)$ is $true$ in

$M^2$ (otherwise either $a$ or $na$ would belong to $M^2$). Hence, there is some $\lambda_j = 1$ and so there is a clause

$$C = a \vee \underbrace{a_1 \vee \ldots \vee a_k}_{H} \leftarrow \underbrace{b_1, \ldots, b_m, not\ c_1, \ldots, not\ c_n}_{B}$$

in $P$ for which $\tilde{\mathcal{U}}_a(H,B)$ is *true* in $M^2$, which means that $\mathcal{V}_M(B) =$ *unknown*, $\mathcal{V}_M(H) =$ *false* and $M^2$ satisfies the constraints $\{a > b_i : 1 \leq i \leq m\}$.

Therefore $M$ is a well-supported model of $P$.

"$\Rightarrow$" Assume that $M$ is a well-supported model of $P$. First, we show that $M^2$ is a model of $P^{3S}$. Since $M$ is a 3-valued model, $M^2$ satisfies all the clauses of the form $a \vee ua \vee na$ and also satisfies the denial rules $IC_P$. Let $a \in \mathcal{L}$ and let the definition of $a$ in $P$ consist of the following set of clauses:

$$\begin{array}{lccc} (C_1) & a \vee H_1 & \leftarrow & B_1 \\ & & \vdots & \\ (C_r) & a \vee H_r & \leftarrow & B_r \end{array}$$

where $r \geq 0$. The 3S–transformation of the definition of $a$ is given by the following set of clauses:

$$\begin{array}{ll} \text{If } r = 0: & \\ & na \leftarrow \\ \text{If } r > 0: & \\ (t_a1) & \quad a \leftarrow \tilde{\mathcal{T}}_a(H_1, B_1) \\ & \quad \vdots \\ (t_ar) & \quad a \leftarrow \tilde{\mathcal{T}}_a(H_r, B_r) \\ (u_a) & ua \leftarrow \{(\tilde{\mathcal{F}}_a/\tilde{\mathcal{U}}_a)^{\lambda_1}(H_1, B_1), \ldots, (\tilde{\mathcal{F}}_a/\tilde{\mathcal{U}}_a)^{\lambda_r}(H_r, B_r) : \\ & \quad \langle \lambda_1, \ldots, \lambda_r \rangle \in \mathcal{B}^r\} \\ (f_a) & na \leftarrow \tilde{\mathcal{F}}_a(H_1, B_1), \ldots, \tilde{\mathcal{F}}_a(H_r, B_r) \\ (3v_a) & a \vee ua \vee na \leftarrow \end{array}$$

Notice that if $r = 0$, there is no support for $a$ to be *true* or *unknown* so $a$ must be *false* in $M$ and then $M^2$ is a model of $na \leftarrow$. To prove the statement when $r > 0$, we consider three cases corresponding to the three possible truth values of $a$ with respect to $M$. It is clear that in each of these cases the clause $(3v_a)$ is satisfied by $M$.

**Case 1:** $a$ is *true* in $M$.

Clearly, $M^2$ models $(t_a1), \ldots, (t_ar)$. Since $M$ is a well-supported model of $P$ there is some $j$, $1 \leq j \leq r$ for which the clause

$$C_j = a \vee \underbrace{a_1 \vee \ldots \vee a_k}_{H_j} \leftarrow \underbrace{b_1, \ldots, b_m, not\ c_1, \ldots, not\ c_n}_{B_j}$$

in $P$ satisfies the following conditions:

1. $a > b_i$ for all $i \in \{1, \ldots, m\}$.
2. $\mathcal{V}_M(B_j) = \mathit{true}$ and $\mathcal{V}_M(H_j) <_t \mathit{true}$.

Therefore $\mathcal{V}_M(\tilde{\mathcal{T}}_a(H_j, B_j)) = \mathit{true}$ and consequently $\mathcal{V}_M(\tilde{\mathcal{F}}_a(H_j, B_j)) = \mathcal{V}_M(\tilde{\mathcal{U}}_a(H_j, B_j)) = \mathit{false}$ and so $M^2$ models $(f_a)$ and $(u_a)$.

**Case 2:** $a$ is *unknown* in $M$.

Clearly, $M^2$ models $(u_a)$. Notice that there is no clause in the definition of $a$ in $P$ for which $\mathcal{V}_M(B_i) = \mathit{true}$ and $\mathcal{V}_M(H_i) < \mathit{true}$. Otherwise $a$ would have to be *true* in $M$ in order for $M$ to be a model of $P$. Hence, for every $i \in \{1, \ldots, r\}$, either $\mathcal{V}_M(\tilde{\mathcal{U}}_a(H_i, B_i)) = \mathit{true}$ or $\mathcal{V}_M(\tilde{\mathcal{F}}_a(H_i, B_i)) = \mathit{true}$ and so, $M^2$ models $(t_a 1), \ldots, (t_a r)$.

Since $a$ is *unknown* in $M$ and $M$ is a well-supported model of $P$ there is a clause

$$C_j = a \vee \underbrace{a_1 \vee \ldots \vee a_k}_{H_j} \leftarrow \underbrace{b_1, \ldots, b_m, not\ c_1, \ldots, not\ c_n}_{B_j}$$

in $P$ satisfying the following conditions:

1. $a > b_i$ for all $i \in \{1, \ldots, m\}$.
2. $\mathcal{V}_M(B_j) = \mathit{unknown}$ and $\mathcal{V}_M(H_j) = \mathit{false}$.

and therefore $\mathcal{V}_M(\tilde{\mathcal{U}}_a(H_j, B_j)) = \mathit{true}$ which implies that $\mathcal{V}_M(\tilde{\mathcal{F}}_a(H_j, B_j)) = \mathit{false}$ and so $M^2$ models $(f_a)$.

**Case 3:** $a$ is *false* in $M$.

Clearly, $M^2$ models $(f_a)$. Since $a$ is *false* in $M$ and $M$ is a model of $P$, $\mathcal{V}_M(B_j) \leq_t \mathcal{V}_M(H_j)$ for all $j, 1 \leq j \leq r$. This implies that $M^2$ models $(t_a 1), \ldots, (t_a r)$ and also that $\mathcal{V}_M(\tilde{\mathcal{U}}_a(H_j, B_j)) = \mathit{false}$ for all $j$ and so $M^2$ models $(u_a)$.

Hence, $M^2$ is a model of $P^{3S}$. It remains to be shown that $M^2$ is a *minimal* model of $P^{3S}$. Suppose, by way of contradiction, that there is some $N \in \mathcal{M}_{P^{3S}}^{IC_P}$ such that $N$ is smaller than $M^2$. It is straightforward to check that $N^3$ is a 3-valued model of $P$ and that $N^3 \prec_t M$. This yields a contradiction since $M$ is a $\prec_t$-minimal 3-valued model of $P$ due to Proposition 2.1 together with the assumption that $M$ is a well-supported (and hence, partial stable) model of $P$.

□

**Corollary 4.2.**
*Let $P$ be an edlpand let $M$ be a 3-valued interpretation of $P$. Then $M$ is a 3-valued stable model of $P$ iff $M^2 \in \mathcal{M}_{P^{3S}}^{IC_P}$.*

*Proof.* This follows immediately from Theorems 3.5 and 4.1 □

1. $\mathcal{M}' := \{\emptyset\}$
2. **repeat**
3. $\quad \mathcal{M} := \mathcal{M}'$
4. $\quad \mathcal{M}' := \emptyset$
5. $\quad$ **for** each $I \in \mathcal{M}$ **do**
6. $\quad\quad I := T_{P_H^{3S}} \uparrow^{\infty} (I)$
7. $\quad\quad$ **if** $I$ satisfies $IC_P$ **then**
8. $\quad\quad\quad$ **if** there is some $C = a \vee ua \vee na \in P_D^{3S}$ s.t. $I \not\models C$ **then**
9. $\quad\quad\quad\quad \mathcal{M}' := \mathcal{M}' \cup \{I \cup \{a\}, I \cup \{ua\}, I \cup \{na\}\}$
10. $\quad\quad\quad$ **else** $\mathcal{M}' := \mathcal{M}' \cup \{I\}$
11. **until** $\mathcal{M} = \mathcal{M}'$
12. $\mathcal{M} := min(\mathcal{M})$
13. $\mathcal{M}_{P^{3S}}^{IC_P} := \{M \in \mathcal{M} : M$ satisfies the constraints in $\mathcal{C}^M\}$

Figure 1: Algorithm to compute $\mathcal{M}_{P^{3S}}^{IC_P}$

## 4.2 Computing Minimal 2-valued Models of $P^{3S}$

In this section we give an algorithm to construct the minimal models of $P^{3S}$ and show how to check which of those models satisfy the constraints in the program.

We start by noticing that $P^{3S}$ contains two types of clauses: Horn clauses and disjunctive facts. Let $P_H^{3S}$ and $P_D^{3S}$ denote the subsets of $P^{3S}$ containing respectively the Horn clauses and the disjunctive facts in $P$.

An approach to compute the minimal 2-valued models of $P^{3S}$ is the following: we start with the empty interpretation and apply an immediate consequence operator to $P_H^{3S}$ until a fixpoint $I$ is reached. If $I$ satisfies all the clauses in $P_D^{3S}$ we are done. Otherwise, we select one clause $a \vee ua \vee na \in P_D^{3S}$ that is not satisfied by $I$ and split $I$ into three interpretations: $I \cup \{a\}$, $I \cup \{ua\}$, and $I \cup \{na\}$. For each such interpretation we apply again the immediate consequence operator with respect to $P_H^{3S}$ to find a revised fixed point, which is tested to determine if it models $P_D^{3S}$. If it does, we are done and if not, we repeat the process until all interpretations obtained satisfy every disjunction in $P_D^{3S}$. If at any point during this process an interpretation inconsistent with the set of denial rules $IC_P$ is reached, then that interpretation is thrown away. At the end of the process, we check which of the resulting interpretations satisfy their own set of constraints.

Figure 1 provides an algorithm to compute $\mathcal{M}_{P^{3S}}^{IC_P}$, where $T_{P_H^{3S}}$ is any immediate consequence operator defined for Horn programs.

We detail now instruction 6 to show how the set of constraints $\mathcal{C}^I$ associated with an interpretation $I$ can be computed simultaneously to the iterations of the fixpoint operator.

```
6.1.  repeat
6.2.     I_0 := I
6.3.     I' := ∅
6.4.     for each clause x ← B under constraints C_x(B) in P_H^{3S}
.        such that V_I(B) = true do
6.5.        I' := I' ∪ {x}
6.6.        if x is of the form a or ua for some a ∈ L then
6.7.           C_a^I := C_a^I ∪ {C_x(B)}
6.8.     I := I'
6.9.  until I = I_0
```

We point out that instruction 13 can be implemented in terms of a search in a particular graph. It is easy to see that a set of constraints $\mathcal{C}$ of the form $a > b$ defines a partial order on $\hat{\mathcal{L}}$ if and only if the directed graph $G = \langle V = \hat{\mathcal{L}}, E = \{(a,b) : a > b \in \mathcal{C}\}\rangle$ is acyclic. Checking if this graph is acyclic can be done in time $O(|\hat{\mathcal{L}}| + |\mathcal{C}|)$ (see e.g. [AHU83]).

We illustrate how the algorithm works for different *edlps*.

**Example 4.4.** *Let* $P = \{a \leftarrow not\ b; b \leftarrow not\ a\}$.

$$
P^{3S} = \{ \quad
\begin{array}{rcll}
a & \leftarrow & nb & \mathcal{C}_a = \emptyset \\
ua & \leftarrow & ub & \mathcal{C}_a = \emptyset \\
na & \leftarrow & b & \qquad\qquad P_H^{3S} \\
b & \leftarrow & na & \mathcal{C}_b = \emptyset \\
ub & \leftarrow & ua & \mathcal{C}_b = \emptyset \\
nb & \leftarrow & a & \\
\hline
a \vee ua \vee na & \leftarrow & & \\
b \vee ub \vee nb & \leftarrow & & \qquad\qquad P_D^{3S}\ \}
\end{array}
$$

$IC_P = \{\Leftarrow x, ux; \Leftarrow x, nx; \Leftarrow ux, nx : x \in \{a, b\}\}$

*We start with the empty interpretation* $I = \{\}$. *Since the Horn part of* $P^{3S}$ *has no facts, the empty set is the fixed point obtained for I. We then select* $a \vee ua \vee na$ *from* $P_D^{3S}$, *which is not satisfied by I and form three interpretations, as shown by the first level of the tree of Figure 2. We find the fixpoint for* $I_1 = \{a\}$ *with respect to* $P_H^{3S}$ *to obtain nb (on the second level of the tree). The interpretation* $I'_1 = \{a, nb\}$ *now satisfies all the clauses in* $P^{3S}$. *The same is done for* $I_2 = \{ua\}$ *and for* $I_3 = \{na\}$ *to obtain ub and b respectively.* $I'_2 = \{ua, ub\}$ *and* $I'_3 = \{na, b\}$ *satisfy all clauses in* $P^{3S}$. *Notice that the three interpretations satisfy the denial rules* $IC_P$ *and they are* $\prec_t$*-incomparable so each of them is* $\prec_t$*-minimal. Finally, each of them satisfies the associated set of constraints:* $\mathcal{C}^{I'_1} = \{\mathcal{C}_a^{I'_1} = \{\emptyset\}\}$, $\mathcal{C}^{I'_2} = \{\mathcal{C}_a^{I'_2} = \{\emptyset\}, \mathcal{C}_b^{I'_2} = \{\emptyset\}\}$ *and* $\mathcal{C}^{I'_3} = \{\mathcal{C}_a^{I'_3} = \{\emptyset\}\}$. *Therefore,* $\mathcal{M}_{P^{3S}}^{IC_P} = \{\{a, nb\}, \{ua, ub\}, \{na, b\}\}$.

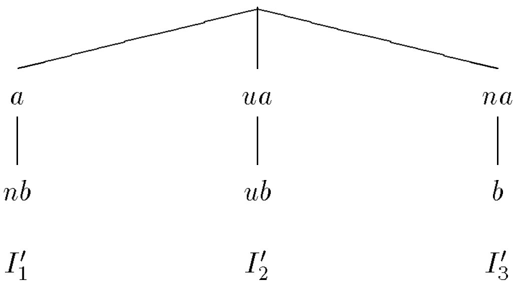



*Figure 2: Minimal 2-valued models of $P^{3S}$ in Example 4.4.*

*The corresponding 3-valued stable models of the program $P$ are $\{\langle\{a\};\{b\}\rangle, \langle\emptyset;\emptyset\rangle, \langle\{b\};\{a\}\rangle\}$. $I'_2$ corresponds to the well-founded model $\langle\emptyset;\emptyset\rangle$ of $P$.*

**Example 4.5.** *Let $P = \{a \leftarrow c; a \leftarrow b; b \leftarrow a; c \leftarrow\}$.*

$$
P^{3S} = \{ \quad
\begin{array}{rcll}
a & \leftarrow & c & \mathcal{C}_a = \{a > c\} \\
a & \leftarrow & b & \mathcal{C}_a = \{a > b\} \\
ua & \leftarrow & uc, nb & \mathcal{C}_a = \{a > c\} \\
ua & \leftarrow & nc, ub & \mathcal{C}_a = \{a > b\} \\
ua & \leftarrow & uc, ub & \mathcal{C}_a = \{a > b, a > c\} \\
na & \leftarrow & nc, nb & \\
b & \leftarrow & a & \mathcal{C}_b = \{b > a\} \\
ub & \leftarrow & ua & \mathcal{C}_b = \{b > a\} \\
nb & \leftarrow & na & \\
c & \leftarrow & & \mathcal{C}_c = \emptyset \\
a \vee ua \vee na & \leftarrow & & \\
b \vee ub \vee nb & \leftarrow & & \\
c \vee uc \vee nc & \leftarrow & & \quad\quad \}
\end{array}
$$

$IC_P = \{\Leftarrow x, ux; \Leftarrow x, nx; \Leftarrow ux, nx : x \in \{a, b, c\}\}$

*The immediate consequence operator applied to $P_H^{3S}$ produces the only model of $P^{3S}$, namely $M = \{a, b, c\}$. The set of constraints associated with $M$ is $\mathcal{C}^M = \{\mathcal{C}_a^M = \{\{a > c\}|\{a > b\}\}, \mathcal{C}_b^M = \{\{b > a\}\}, \mathcal{C}_a^M = \{\emptyset\}\}$. $M$ satisfies this set of constraints since $\{a > c, b > a\}$ is a partial order on $\{a, b, c\}$. Hence, $\mathcal{M}_{P^{3S}}^{IC_P} = \{\{a, b, c\}\}$ and 3-STABLE$(P) = \langle\{a, b, c\};\emptyset\rangle$.*

The algorithm in Figure 1 constructs every minimal model of $P^{3S}$ and hence, in the worst case, runs in exponential time on the size of $P^{3S}$.

A global improvement to the process of computing the partial stable models of $P$ is to partition $P$ into several connected components using the notion of semi-stratification described in [FLMS93] and to apply the 3S–transformation and the algorithm in Figure 1 just to each component of the program.

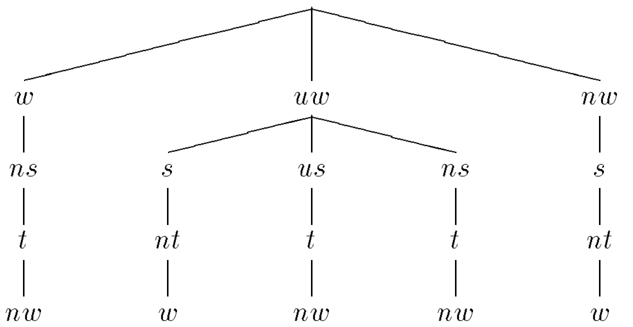



*Figure 3: Computation of the minimal 2-valued models of $P^{3S}$ in Example 4.6.*

A local speed–up in the algorithm can be achieved by selecting in instruction 8, a clause from $P_D^{3S}$ that maximizes the number of clauses in $P_H^{3S}$ that are usable in the next application of the fixpoint operator $T_{P_H^{3S}}$.

We end this section by showing how the algorithm works with a program that does not have any partial stable models.

**Example 4.6.** *Let* $P = \{w \vee s \vee t; w \leftarrow not\ t; s \leftarrow not\ w; t \leftarrow not\ s\}$.

$$
\begin{array}{llrcll}
P^{3S} = \{ & & w & \leftarrow & (us|ns), (ut|nt) & \mathcal{C}_w = \emptyset \\
& & w & \leftarrow & nt & \mathcal{C}_w = \emptyset \\
& & uw & \leftarrow & (s|t), ut & \mathcal{C}_w = \emptyset \\
& & nw & \leftarrow & (s|t), t & \\
& & s & \leftarrow & (uw|nw), (ut|nt) & \mathcal{C}_s = \emptyset \\
& & s & \leftarrow & nw & \mathcal{C}_s = \emptyset \\
& & us & \leftarrow & (w|t), uw & \mathcal{C}_s = \emptyset \\
& & ns & \leftarrow & (w|t), w & \\
& & t & \leftarrow & (uw|nw), (us|ns) & \mathcal{C}_t = \emptyset \\
& & t & \leftarrow & ns & \mathcal{C}_t = \emptyset \\
& & ut & \leftarrow & (w|s), us & \mathcal{C}_t = \emptyset \\
& & nt & \leftarrow & (w|s), s & \\
& & w \vee uw \vee nw & \leftarrow & & \\
& & s \vee us \vee ns & \leftarrow & & \\
& & t \vee ut \vee nt & \leftarrow & & \}
\end{array}
$$

$IC_P = \{\Leftarrow x, ux; \Leftarrow x, nx; \Leftarrow ux, nx : x \in \{w, s, t\}\}$

*Figure 3 shows that every interpretation obtained during the computation of the minimal 2-valued models of $P^{3S}$ is inconsistent with respect to the denial rules $IC_P$. Then $\mathcal{M}_{P^{3S}}^{IC_P} = \{\}$ and consequently* 3-STABLE*(P)* $= \{\}$.

## 5 Conclusions

We have provided an effective procedure that computes the partial stable models of an *edlp*. We have shown that there is a one-to-one correspondence between the partial stable models of an *edlp* and the minimal models of a constrained *edlp* free of negation-by-default (or equivalently, the well-supported models of an *edlp* free of negation-by-default). Strictly speaking, this implies that the use of negation-by-default under the interpretation of the partial stable model semantic does not increase the expressive power of constrained positive programs. The same observation is applicable to the (total) stable model and the well-founded semantics since these semantics are easily derived from the set of partial stable models of the program.

Nevertheless, the presence of the negation-by-default operator is undoubtly useful in the sense that it allows us to write concise programs independent of the number of truth values being considered.

The procedure presented here to compute the 3-valued stable models of an *edlp* is based on case analysis. An implementation of that procedure has been completed and we expect to experiment with it. We believe that the approach can be adapted to compute the 2-valued as well as the 4-valued stable models [Fit93] of the program. We plan to investigate these topics.

## References


[AHU83] A. Aho, J. Hopcroft, and J. Ullman. *Data Structures and Algorithms*. Addison-Wesley, 1983.

[BNNS93] C. Bell, A. Nerode, R. Ng, and V.S. Subrahmanian. Implementing stable model semantics by linear programming. In *Proceedings of the 1993 International Workshop on Logic Programming and Non-monotonic Reasoning*, June 1993.

[EG93] T. Eiter and G. Gottlob. Complexity aspects of various semantics for disjunctive databases. In *Proceedings of the Twelfth ACM SIGART–SIGMOD–SIGART Symposium on Principles of Database Systems (PODS-93)*, pages 158–167. ACM Press, May 1993.

[Fag91] F. Fages. A new fixpoint semantics for general logic programs compared with the well-founded and the stable model semantics. *New Generation Computing*, 9:425–443, 1991.

[Fit85] M. Fitting. A Kripke-Kleene semantics of logic programs. *Journal of Logic Programming*, 2(4):295–312, December 1985.

[Fit93] M. Fitting. The family of stable models. *Journal of Logic Programming*, 17(2, 3 & 4):197–226, 1993.

[FLMS93] J.A. Fernández, J. Lobo, J. Minker, and V.S. Subrahmanian. Disjunctive lp + integrity constrains = stable model semantics. *Annals of Mathematics and Artificial Intelligence*, 8(3-4):449–474, 1993.

[GL88] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In R. Kowalski and K. Bowen, editors, *Proceedings of the Fifth Inter-*

*national Conference and Symposium on Logic Programming*, pages 1070–1080, Seattle, WA, USA, Aug. 1988. The MIT Press.

[IKH92] K. Inoue, M. Koshimura, and R. Hasegawa. Embedding negation as failure into a model generation theorem prover. In D. Kapur, editor, *Proceedings of the Eleventh International Conference on Automated Deduction*, pages 400–415, Saratoga Springs NY, USA, June 1992. Springer-Verlag.

[MR90] J. Minker and A. Rajasekar. A fixpoint semantics for disjunctive logic programs. *Journal of Logic Programming*, 9(1):45–74, July 1990.

[MR93] J. Minker and C. Ruiz. On extended disjunctive logic programs. In J. Komorowski and Z.W. Raś, editors, *Proceedings of the Seventh International Symposium on Methodologies for Intelligent Systems*, pages 1–18. Lecture Notes in AI. Springer-Verlag, June 1993. (Invited Paper).

[MR94] J. Minker and C. Ruiz. Semantics for disjunctive logic programs with explicit and default negation. *Fundamenta Informaticae*, 20(3/4):145–192, 1994. Anniversary Issue edited by H. Rasiowa.

[Prz91] T. C. Przymusinski. Stable semantics for disjunctive programs. *New Generation Computing*, 9:401–424, 1991.