An ASP Approach to Generate Minimal Countermodels in Intuitionistic Propositional Logic

Camillo Fiorentini
Department of Computer Science, University of Milan, Italy
fiorentini@di.unimi.it

Abstract

Intuitionistic Propositional Logic is complete w.r.t. Kripke semantics: if a formula is not intuitionistically valid, then there exists a finite Kripke model falsifying it. The problem of obtaining concise models has been scarcely investigated in the literature. We present a procedure to generate minimal models in the number of worlds relying on Answer Set Programming (ASP).

1 Introduction

Intuitionistic Propositional Logic (IPL) has the finite model property w.r.t. Kripke semantics: if a formula G is not valid in IPL, then G can be falsified in a finite (Kripke) model $\mathcal{K}$; we call $\mathcal{K}$ a *countermodel* for G. In many of the proof-search strategies for IPL (see [Dyckhoff, 2016] for a review), whenever the search for a derivation of a formula G fails, a countermodel for G is provided. A countermodel can be understood as a "certificate" witnessing the non-validity of the formula G, thus countermodels can be used for diagnosis, to analyze why some property fails or to fix errors in formal specifications. This methodology has been successfully applied to Property-Based Testing (PBT), where the user defines a property that the code to be checked is expected to fulfill and the PBT engine generates test cases to validate it; failing cases (countermodels) detect flaws and are exploited to amend the code (see, e.g., [Cheney *et al.*, 2016] for an application to αProlog). In this perspective, it is critical that countermodels are minimal so as to convey a plain and concise representation of non-validity. For classical logics, various notions of minimality have been introduced, aiming at reducing the domain of interpretations (e.g. [Lorenz, 1994]) or the interpretation of predicates (e.g. [Niemelä, 1996; Bry and Yahya, 2000]). For modal and description logics, a semantic notion of minimality based on subset-simulation has been proposed in [Papacchini and Schmidt, 2011; 2014], where terminating procedures to build minimal models for some of the logics between K and S5 are presented.

Here we aim at minimizing the size of countermodels, measured by the number of worlds: a countermodel $\mathcal{K}$ for G is *minimal* if no countermodel for G has size less then the size of $\mathcal{K}$ (thus, if G is not classically valid, the minimal countermodel only contains a world). This issue has been scarcely investigated in the literature, where the emphasis is mainly on derivations, and countermodels are often obtained as a byproduct (see, e.g., [Pinto and Dyckhoff, 1995; Corsi and Tassi, 2007; Negri, 2014; Claessen and Rosén, 2015]). Also the known refutation calculi for IPL, namely the calculi designed to prove the non-validity of formulas (see, e.g., [Skura, 2011]), do not focus on the size of countermodels. More relevant to this issue are the proof-search procedures described in [Galmiche and Larchey-Wendling, 1999; Larchey-Wendling *et al.*, 2001; Svejdar, 2006; Ferrari *et al.*, 2013; 2015; Fiorentini and Ferrari, 2017], which yield models having depth bounded by the goal formula; however, minimality of countermodels is not ensured. One might naively guess that it is always possible to shrink a given countermodel, for instance by using the filtration techniques discussed in [Chagrov and Zakharyaschev, 1997]; actually, it is unlikely that such a general method exists (see Sec. 3).

To tackle the problem, we propose a quite different strategy, based on model generation: given a goal formula G, we try to build a countermodel for G by a model-search procedure guided by semantics. A naive implementation of the process immediately blows-up and, even for small goal formulas, model generation is not terminating; thus, we need a clever formalization of the problem. Following [Goré and Thomson, 2012; Goré *et al.*, 2014], worlds of models are represented by sets $\mathcal{W}$ of *atomic* subformulas H of G (namely: H is either a propositional variable or $H = \neg A$ or $H = A \to B$) satisfying some closure properties. The first selected set $\mathcal{W}$ is a putative world falsifying G. To get a well-defined Kripke model, we have to guarantee that atomic subformulas of G not belonging to $\mathcal{W}$ are not valid in $\mathcal{W}$. For instance, if $A \to B \notin \mathcal{W}$, we need a world $\mathcal{Z}$ such that $\mathcal{Z}$ is a successor of $\mathcal{W}$ (namely, $\mathcal{W} \subseteq \mathcal{Z}$) and $\mathcal{Z}$ witnesses the non-validity of $A \to B$ in $\mathcal{W}$ (namely, A holds in $\mathcal{Z}$ and B does not hold). This triggers a saturation process which successfully ends when all the needed witnesses have been generated, thus yielding a countermodel for G.

We formalize the problem in Answer Set Programming (ASP) [Baral, 2010], a form of declarative programming based on the stable model semantics (answer sets), which enables to solve hard search problems in a uniform way [Dantsin *et al.*, 2001]. We define an ASP program Π_G such that an answer set of Π_G corresponds to a countermodel for G; if no answer exists, there is no countermodel for G,

meaning that G is valid (in IPL). To compute answer sets, we exploit the Potassco tool `clingo` [Gebser *et al.*, 2011]. The minimization of models is delegated to `clingo`; however, it is critical to encode the problem so that even the first computed model is small, otherwise the minimization engine gets stuck. Differently from other declarative formalisms, ASP allows for a quite modular formalization; as outlined in Sec. 4, the generator can be easily extended to deal with intermediate logics where the frame conditions can be expressed in ASP, such as Gödel-Dummett logic [Dummett, 1959] and Here and There logic [Pearce, 1997]. The implementation is available at author's home page.

2 Minimal Countermodels in IPL

We consider the propositional language $\mathcal{L}$ based on a denumerable set of propositional variables $\mathcal{V}$ and the connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$. Let A be a formula. By $\mathrm{Sf}(A)$ we denote the set of all subformulas of A. The *size* of A, denoted by $|A|$, is the number of symbols occurring in A. A *(Kripke) model* is a triple $\langle P, \leq, V\rangle$, where $\langle P, \leq\rangle$ is a finite poset and V (the evaluation function) is a monotone map $P \rightarrow 2^{\mathcal{V}}$, namely: $w \leq w'$ implies $V(w) \subseteq V(w')$; elements of P are called *worlds*. The *forcing relation* $\Vdash \subseteq P \times \mathcal{L}$ is defined as usual:

$$
\begin{array}{lll}
w \Vdash p & \text{iff} & p \in V(w), \text{ where } p \in \mathcal{V} \\
w \Vdash \neg A & \text{iff} & \forall w' \geq w,\ w' \nVdash A \\
w \Vdash A \wedge B & \text{iff} & w \Vdash A \text{ and } w \Vdash B \\
w \Vdash A \vee B & \text{iff} & w \Vdash A \text{ or } w \Vdash B \\
w \Vdash A \rightarrow B & \text{iff} & \forall w' \geq w,\ w' \Vdash A \text{ implies } w' \Vdash B.
\end{array}
$$

Note that $\Vdash$ is monotone, namely: $w \Vdash A$ and $w \leq w'$ implies $w' \Vdash A$. If Γ is a set of formulas, by $w \Vdash \Gamma$ we mean $w \Vdash A$ for every $A \in \Gamma$. A formula G is valid in IPL iff, for every model $\mathcal{K}$ and world w of $\mathcal{K}$, $w \Vdash G$. Thus, to certify that G is not valid (in IPL) we can exhibit a model $\mathcal{K}$ containing a world w such that $w \nVdash G$; we call $\mathcal{K}$ a *countermodel for* G. We aim at representing worlds of $\mathcal{K}$ by means of proper subsets $\mathcal{W}$ of $\mathrm{Sf}(G)$. To avoid redundancies, we consider sets $\mathcal{W}$ only containing subformulas H of G of the form $H \in \mathcal{V}$ or $H = \neg A$ or $H = A \rightarrow B$; we call H an *atomic subformula* of G and by $\mathrm{At}(G)$ we denote the set of all atomic subformulas of G. By $\mathrm{Cl}_G(\mathcal{W})$ we denote the smallest set satisfying the following properties, where $-(\neg A) = A$ and $-A = \neg A$ if $A \neq \neg B$:

- $\mathcal{W} \subseteq \mathrm{Cl}_G(\mathcal{W}) \subseteq \mathrm{Sf}(G)$.
- $C = A \wedge B \in \mathrm{Sf}(G)$ and $\{A, B\} \subseteq \mathrm{Cl}_G(\mathcal{W})$ implies $C \in \mathrm{Cl}_G(\mathcal{W})$.
- $C = A \vee B \in \mathrm{Sf}(G)$ and $\{A, B\} \cap \mathrm{Cl}_G(\mathcal{W}) \neq \emptyset$ implies $C \in \mathrm{Cl}_G(\mathcal{W})$.
- $C = A \rightarrow B \in \mathrm{Sf}(G)$ and $\{-A, B\} \cap \mathrm{Cl}_G(\mathcal{W}) \neq \emptyset$ implies $C \in \mathrm{Cl}_G(\mathcal{W})$.
- $C = \neg\neg A \in \mathrm{Sf}(G)$ and $A \in \mathrm{Cl}_G(\mathcal{W})$ implies $C \in \mathrm{Cl}_G(\mathcal{W})$.
- $C = \neg(A \wedge B) \in \mathrm{Sf}(G)$ and $\{-A, -B\} \cap \mathrm{Cl}_G(\mathcal{W}) \neq \emptyset$ implies $C \in \mathrm{Cl}_G(\mathcal{W})$.
- $C = \neg(A \vee B) \in \mathrm{Sf}(G)$ and $\{-A, -B\} \subseteq \mathrm{Cl}_G(\mathcal{W})$ implies $C \in \mathrm{Cl}_G(\mathcal{W})$.

$T = S \rightarrow X \vee \neg\neg p$ where:
$S = (X \rightarrow D) \rightarrow D \quad X = \neg\neg p \rightarrow p \quad D = \neg\neg p \vee \neg p$
$\mathrm{At}(T) = \{\, p,\ \neg p,\ \neg\neg p,\ X,\ X \rightarrow D,\ S,\ T \,\}$

$$
\begin{array}{ll}
\mathcal{W}_0 = \emptyset & \mathcal{W}_5 = \{\, X \rightarrow D,\ T \,\} \\
\mathcal{W}_1 = \{X \rightarrow D\} & \mathcal{W}_6 = \{X,\ S,\ T\} \\
\mathcal{W}_2 = \{S\} & \mathcal{W}_7 = \{\, \neg\neg p,\ X \rightarrow D,\ S,\ T \,\} \\
\mathcal{W}_3 = \{T\} & \mathcal{W}_8 = \{\, \neg p,\ X,\ X \rightarrow D,\ S,\ T \,\} \\
\mathcal{W}_4 = \{X,\ T\} & \mathcal{W}_9 = \{\, p,\ \neg\neg p,\ X,\ X \rightarrow D,\ S,\ T \,\}
\end{array}
$$

Figure 1: The formula T and the related p-worlds $\mathcal{W}_0, \ldots, \mathcal{W}_9$

- $C = \neg(A \rightarrow B) \in \mathrm{Sf}(G)$ and $\{A, -B\} \subseteq \mathrm{Cl}_G(\mathcal{W})$ implies $C \in \mathrm{Cl}_G(\mathcal{W})$.

By induction on formulas, we can prove that:

(P1) $\mathcal{W} \subseteq \mathcal{Z}$ implies $\mathrm{Cl}_G(\mathcal{W}) \subseteq \mathrm{Cl}_G(\mathcal{Z})$.

(P2) $w \Vdash \mathcal{W}$ implies $w \Vdash \mathrm{Cl}_G(\mathcal{W})$.

A set $\mathcal{W} \subseteq \mathrm{At}(G)$ is a *p-world (possible world)* iff it satisfies the following closure properties:

$$
\begin{array}{ll}
(\mathrm{C}\neg) & \neg A \in \mathcal{W} \Longrightarrow A \notin \mathrm{Cl}_G(\mathcal{W}) \\
(\mathrm{C}\rightarrow) & (A \rightarrow B \in \mathcal{W}) \wedge (A \in \mathrm{Cl}_G(\mathcal{W})) \Longrightarrow B \in \mathrm{Cl}_G(\mathcal{W}) \\
(\mathrm{CAt}) & H \in \mathrm{At}(G) \cap \mathrm{Cl}_G(\mathcal{W}) \Longrightarrow H \in \mathcal{W}.
\end{array}
$$

In the ASP representation, we limit ourselves to use p-worlds as candidate worlds of countermodels; as shown in the examples, this considerably reduces the number of subsets of $\mathrm{At}(G)$ to be considered in model generation.

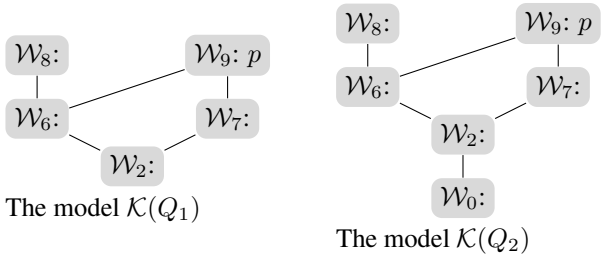
Example 1 Let T be the non-valid formula defined in Fig. 1 (T is equivalent to Nishimura formula N_9, see Ex. 7). Then:

- $\mathcal{Z}_0 = \mathrm{At}(T)$ satisfies the closure properties $(\mathrm{C}\rightarrow)$ and (CAt), but not $(\mathrm{C}\neg)$, since $\neg\neg p \in \mathcal{Z}_0$ and $\neg p \in \mathrm{Cl}_T(\mathcal{Z}_0)$.
- $\mathcal{Z}_1 = \{S, T\}$ satisfies $(\mathrm{C}\neg)$ and (CAt), but not $(\mathrm{C}\rightarrow)$, since $T \in \mathcal{Z}_1$, $S \in \mathrm{Cl}_T(\mathcal{Z}_1)$ and $X \vee \neg\neg p \notin \mathrm{Cl}_T(\mathcal{Z}_1)$.
- Let $\mathcal{Z}_2 = \{\neg p, X \rightarrow D, S, T\}$. Since $-(\neg\neg p) = \neg p \in \mathcal{Z}_2$, then $X \in \mathrm{Cl}_T(\mathcal{Z}_2)$. This implies that $\mathcal{Z}_2$ satisfies $(\mathrm{C}\neg)$ and $(\mathrm{C}\rightarrow)$ but not (CAt).

The set $\mathrm{At}(T)$ contains 7 elements, giving rise to $2^7 = 128$ subsets, but only 10 of them are p-worlds, namely the sets $\mathcal{W}_0, \ldots, \mathcal{W}_9$ displayed in Fig. 1. $\diamond$

This representation is close to [Goré and Thomson, 2012], with some tweak. We treat $\neg$ as a primitive connective (thus, $\neg p$ is not an abbreviation for $p \rightarrow \bot$) and, in the definition of $\mathrm{Cl}_G(\mathcal{W})$, negated formulas are analyzed according to the constructive interpretation of $\neg$. This reduces the number of p-worlds; for instance, in Ex. 1 we keep out p-worlds $\mathcal{W}$ such that $p \in \mathcal{W}$ and $\neg\neg p \notin \mathcal{W}$.

Let P be a set of p-worlds; by $\mathcal{K}(P)$ we denote the model $\langle P, \leq, V\rangle$ such that $\leq$ coincides with the subset relation $\subseteq$ and, for every $\mathcal{W} \in P$, $V(\mathcal{W}) = \mathcal{W} \cap \mathcal{V}$. In general $\mathcal{W}$ does not coincide with the set of atomic subformulas of G forced at $\mathcal{W}$. For instance, it might happen that $A \rightarrow B \notin \mathcal{W}$, but



The model $\mathcal{K}(Q_1)$ The model $\mathcal{K}(Q_2)$

Figure 2: T-saturated sets and the related models

there is no $\mathcal{Z} \in P$ witnessing that $\mathcal{W} \nVdash A \to B$, namely: $\mathcal{W} \leq \mathcal{Z}$ and $\mathcal{Z} \Vdash A$ and $\mathcal{Z} \nVdash B$. To get a close correspondence between membership and forcing we introduce the notion of G-saturation. A set of p-worlds P is *G-saturated* iff, for every $\mathcal{W} \in P$, the following saturation properties hold:

(S1) for every $\neg A \in \mathrm{At}(G) \setminus \mathcal{W}$, there is $\mathcal{Z} \in P$ such that $\mathcal{W} \subseteq \mathcal{Z}$ and $A \in \mathrm{Cl}_G(\mathcal{Z})$.

(S2) For every $A \to B \in \mathrm{At}(G) \setminus \mathcal{W}$, there is $\mathcal{Z} \in P$ such that $\mathcal{W} \subseteq \mathcal{Z}$ and $A \in \mathrm{Cl}_G(\mathcal{Z})$ and $B \notin \mathrm{Cl}_G(\mathcal{Z})$.

Example 2 (Ex. 1 cont) The sets below are T-saturated:

$$Q_1 \;=\; \{\, \mathcal{W}_2, \mathcal{W}_6, \mathcal{W}_7, \mathcal{W}_8, \mathcal{W}_9 \,\} \qquad Q_2 \;=\; Q_1 \cup \{\mathcal{W}_0\}$$

For instance, we have $X = \neg\neg p \to p \notin \mathcal{W}_2$ and (S2) is matched by setting $\mathcal{Z} = \mathcal{W}_7$. The models $\mathcal{K}(Q_1)$ and $\mathcal{K}(Q_2)$ are displayed in Fig. 2; for each world $\mathcal{W}_k$, we list the propositional variables in $\mathcal{W}_k$. In both models, for every $H \in \mathrm{At}(T)$, $\mathcal{W}_k \Vdash H$ iff $H \in \mathcal{W}_k$ (see next lemma), accordingly $\mathcal{W}_2 \nVdash T$. ◇

Lemma 1 *Let P be a G-saturated set. For every $\mathcal{W} \in P$ and $A \in \mathrm{Sf}(G)$, $\mathcal{W} \Vdash A$ in $\mathcal{K}(P)$ iff $A \in \mathrm{Cl}_G(\mathcal{W})$.*

Proof: Let $\mathcal{K}(P) = \langle P, \leq, V\rangle$, $\mathcal{W} \in P$ and $A \in \mathrm{Sf}(G)$; we prove the lemma by induction on $|A|$; we only detail some representative case. Let $A = \neg B$ (thus $\neg B \in \mathrm{At}(G)$). If $\neg B \notin \mathrm{Cl}_G(\mathcal{W})$, then $\neg B \notin \mathcal{W}$ hence, by (S1), there is $\mathcal{Z} \in P$ such that $\mathcal{W} \subseteq \mathcal{Z}$ and $B \in \mathrm{Cl}_G(\mathcal{Z})$. By (IH), $\mathcal{Z} \Vdash B$; since $\mathcal{W} \leq \mathcal{Z}$, we conclude $\mathcal{W} \nVdash \neg B$. Conversely, let $\mathcal{W} \nVdash \neg B$. Then, there is $\mathcal{Z} \in P$ such that $\mathcal{W} \leq \mathcal{Z}$ and $\mathcal{Z} \Vdash B$; by (IH), $B \in \mathrm{Cl}_G(\mathcal{Z})$. Since $\mathcal{Z}$ satisfies $(\mathrm{C}\neg)$, $\neg B \notin \mathcal{Z}$. Since $\mathcal{W} \subseteq \mathcal{Z}$, we get $\neg B \notin \mathcal{W}$ and, by (CAt), we conclude $\neg B \notin \mathrm{Cl}_G(\mathcal{W})$. Let $A = B_0 \vee B_1$. If $\mathcal{W} \Vdash B_0 \vee B_1$, then $\mathcal{W} \Vdash B_k$, with $k \in \{0, 1\}$. By (IH), $B_k \in \mathrm{Cl}_G(\mathcal{W})$, which implies $B_0 \vee B_1 \in \mathrm{Cl}_G(\mathcal{W})$. Conversely, if $B_0 \vee B_1 \in \mathrm{Cl}_G(\mathcal{W})$, then $B_k \in \mathrm{Cl}_G(\mathcal{W})$, with $k \in \{0, 1\}$. By (IH), $\mathcal{W} \Vdash B_k$, hence $\mathcal{W} \Vdash B_0 \vee B_1$. ⧫

Lemma 2 *Let $\mathcal{K} = \langle P, \leq, V\rangle$ be a model and G a formula. Then, there exist a G-saturated set Q and a surjective map $\phi : P \to Q$ such that, for every $w \in P$ and $A \in \mathrm{Sf}(G)$, $w \Vdash A$ iff $A \in \mathrm{Cl}_G(\phi(w))$.*

Proof: Let ϕ be the map from P to $2^{\mathrm{At}(G)}$ defined as

$$\phi(w) \;=\; \{\, H \in \mathrm{At}(G) \mid w \Vdash H \,\}$$

and $w \in P$; we prove that:

(1) for every $A \in \mathrm{Sf}(G)$, $w \Vdash A$ iff $A \in \mathrm{Cl}_G(\phi(w))$.

(2) $\phi(w)$ is a p-world.

Since $w \Vdash \phi(w)$, by (P2) we get $w \Vdash \mathrm{Cl}_G(\phi(w))$, and this proves the if part of (1). Conversely, let $A \in \mathrm{Sf}(G)$ such that $w \Vdash A$; by induction on $|A|$ one can easily show that $A \in \mathrm{Cl}_G(\phi(w))$, and this proves (1). Let $\neg A \in \phi(w)$; then $w \Vdash \neg A$, hence $w \nVdash A$. By (1), $A \notin \mathrm{Cl}_G(\phi(w))$, hence $\phi(w)$ satisfies $(\mathrm{C}\neg)$. The proof that $\phi(w)$ matches both $(\mathrm{C}\to)$ and (CAt) is similar, and this concludes the proof of (2). Let Q be the ϕ-image of P; we check that Q satisfies (S1) (the proof for (S2) is similar). Let $w \in P$ and let $\neg A \in \mathrm{At}(G) \setminus \phi(w)$. By (CAt), $\neg A \notin \mathrm{Cl}_G(\phi(w))$ hence, by (1), $w \nVdash \neg A$. There is $z \in P$ such that $w \leq z$ and $z \Vdash A$; by (1), $A \in \mathrm{Cl}_G(\phi(z))$. Since $\phi(z) \in Q$ and $\phi(w) \subseteq \phi(z)$, (S1) follows. ⧫

A G-saturated set Q is *complete* iff Q contains a p-world $\mathcal{W}$ such that $G \notin \mathrm{Cl}_G(\mathcal{W})$. By $|Q|$ we denote the cardinality of Q; the size $|\mathcal{K}|$ of a model $\mathcal{K}$ is the number of its worlds.

Lemma 3 *Let $\mathcal{K}$ be a countermodel for G. There exists a complete G-saturated set Q such that $|Q| \leq |\mathcal{K}|$.*

Proof: Let Q and ϕ be as asserted in Lemma 2. By hypothesis, there exists a world w of $\mathcal{K}$ such that $w \nVdash G$; this implies $G \notin \mathrm{Cl}_G(\phi(w))$, hence Q is complete. By surjectivity of ϕ, we get $|Q| \leq |\mathcal{K}|$. ⧫

A countermodel $\mathcal{K}$ for G is a *minimal countermodel for G* iff there is no countermodel $\mathcal{K}'$ for G such that $|\mathcal{K}'| < |\mathcal{K}|$. A complete G-saturated set Q is *minimal* if there exists no complete G-saturated set Q' such that $|Q'| < |Q|$. We state the main results of this section:

Theorem 1 *Let G be a formula.*

(1) If $G \notin$ IPL, then there exists a complete G-saturated set.

(2) If Q is a complete G-saturated set, then $\mathcal{K}(Q)$ is a countermodel for G.

(3) If Q is a minimal complete G-saturated set, then $\mathcal{K}(Q)$ is a minimal countermodel for G.

Proof: Point (1) follows by Lemma 3, Point (2) by Lemma 1. Finally, let us assume that Q is a minimal complete G-saturated set and, by absurd, $\mathcal{K}(Q)$ is not minimal. Then, there is a countermodel $\mathcal{K}'$ for G such that $|\mathcal{K}'| < |\mathcal{K}(Q)| = |Q|$. By Lemma 3, there is a complete G-saturated set Q' such that $|Q'| \leq |\mathcal{K}'|$. Thus, $|Q'| < |Q|$, against the hypothesis that Q is minimal, hence $\mathcal{K}(Q)$ is minimal. ⧫

Example 3 (Ex. 2 cont) The T-saturated sets Q_1 and Q_2 are complete ($T \notin \mathcal{W}_2$), hence both $\mathcal{K}(Q_1)$ and $\mathcal{K}(Q_2)$ are countermodels for T. Moreover, Q_1 is minimal, hence $\mathcal{K}(Q_1)$ is a minimal countermodel for T. ◇

By Theorem 1, to build a countermodel for a goal formula G, we have to search for a complete G-saturated set Q; each $\mathcal{W} \in Q$ corresponds to a world of the countermodel and $\mathcal{W} \Vdash A$ iff $A \in \mathrm{Cl}_G(\mathcal{W})$, for every $A \in \mathrm{Sf}(G)$. To get a minimal countermodel, we have to minimize $|Q|$.

3 ASP Implementation

We describe an ASP program Π_G to generate a minimal countermodel for a goal formula G. According with the ASP paradigm (see e.g. [Baral, 2010]), Π_G is a Prolog-like program consisting of two components $\texttt{Goal}(G)$ and $\texttt{Gen}$:

- $\texttt{Goal}(G)$ is a set of ground facts representing the specific instance of the problem related to the goal G;
- $\texttt{Gen}$ encodes the model search algorithm.

A solution to $\Pi_G = \texttt{Gen} \cup \texttt{Goal}(G)$, called *answer set*, corresponds to a countermodel for G; if Π_G has no answers, then no countermodel for G exists, hence G is valid (in IPL). We exploit the ASP solver `clingo` [Gebser *et al.*, 2012]. By Theorem 1, a countermodel is identified by a complete G-saturated set P. Worlds of P are selected from the available p-worlds by applying the choice rules $(\mathrm{Ch_r})$ and $(\mathrm{Ch_w})$. Rule $(\mathrm{Ch_r})$ chooses a p-world w_0 such that $G \notin \mathrm{Cl}_G(w_0)$, which is nominated to be the *root* of the countermodel (namely, the minimal world w.r.t. $\leq$); w_0 will be the world of the countermodel falsifying G. Whenever a new world w is selected, the saturation conditions (S1) and (S2) are checked to test that formulas $\neg A$ and $A \to B$ not belonging to w have the required witnesses; if this is not the case, new worlds are added by the choice rule $(\mathrm{Ch_w})$. If the saturation process successfully ends, we get a G-saturated set, namely a countermodel for G. Then, the minimization engine of `clingo` searches for models having fewer worlds, until an optimum answer is found. We present the relevant sections of the program; we assume that G is the goal formula and $N = |\mathrm{At}(G)|$.

Encoding $\texttt{Goal}(G)$ of the Goal Formula G

To encode G, we introduce the set of new atoms $\mathcal{A} = \{a(0), \ldots, a(N-1)\}$, each corresponding to an atomic subformula of G. We call $\mathcal{L}_\mathcal{A}$ the language based on $\mathcal{A}$, and this is the language used by the generator. The user has to settle a 1-1 map $\psi : \mathcal{A} \to \mathrm{At}(G)$; by $\psi^\star$ we denote the homomorphic extension of ψ to $\mathcal{L}_\mathcal{A}$ [1]. The map ψ is encoded by a set of ground facts using the following predicates, where $a(k) \in \mathcal{A}$, $p \in \mathcal{V}$ and $A \in \mathcal{L}_\mathcal{A}$:

$$\begin{array}{rcl} \texttt{count\_atoms}(M) & \text{iff} & M = |\mathrm{At}(G)| \\ \texttt{at\_to\_PV}(a(k), p) & \text{iff} & \psi(a(k)) = p \\ \texttt{def}(a(k), A) & \text{iff} & \psi(a(k)) = \psi^\star(A) \\ \texttt{goal}(A) & \text{iff} & \psi^\star(A) = G. \end{array}$$

We assume that, for every $0 \leq k \leq N-1$, $\texttt{Goal}(G)$ contains either one definition $\texttt{at\_to\_PV}(a(k), p)$ or one definition $\texttt{def}(a(k), A)$, where in A only atoms $a(j)$ such that $j < k$ occur; in the latter case, we write $a(k) \coloneqq A$.

Example 4 The formula T in Fig. 1 has 7 atomic subformulas, thus $\mathcal{A} = \{a(0), \ldots, a(6)\}$. We set ($\mapsto$ represents $\psi^\star$):

$$\begin{array}{l} a(0) \mapsto p \quad a(1) \coloneqq \neg a(0) \mapsto \neg p \qquad a(2) \coloneqq \neg a(1) \mapsto \neg\neg p \\ a(3) \coloneqq a(2) \to a(0) \mapsto X \\ a(4) \coloneqq a(3) \to a(2) \vee a(1) \mapsto X \to D \\ a(5) \coloneqq a(4) \to a(2) \vee a(1) \mapsto S \\ a(6) \coloneqq a(5) \to a(3) \vee a(2) \mapsto T \qquad \text{// goal} \end{array}$$

This is translated by the following ground facts:

```
count_atms(7).   at_to_PV(a(0),p).
def(a(1),neg(a(0))). def(a(2),neg(a(1))).  ...
def(a(6), imp(a(5),or(a(3),a(2)))). goal(a(6)).
```

◇

Encoding `Gen` of the Model Generator

To represent the 2^N subsets of $\mathcal{A}$, we rely on binary representation of natural numbers. Let $0 \leq k \leq 2^N - 1$ and let $(k)_2 = b_m\, b_{m-1} \cdots b_1\, b_0$ be the binary representation of k; then, $w(k)$ is the set of atoms $a(i)$ such that $b_i = 1$. For instance, $w(0)$ is the empty set and, since $(13)_2 = 1101$, $w(13)$ represents the set $\{a(0), a(2), a(3)\}$. Membership and inclusion can be efficiently implemented, using the built-in arithmetical and logical operators. Indeed, the following properties hold, where ':' is the integer division, '%' the modulo, '$\sim$' the bitwise negation, '|' the bitwise disjunction and $(-1)_2 = 11111\ldots$:

$$\begin{array}{lll} a(i) \in w(k) & \text{iff} & (k : 2^i)\,\%\,2 = 1 \\ w(k) \subseteq w(m) & \text{iff} & (\sim (k)_2) \mid (m)_2 = (-1)_2 \end{array}$$

Properties characterizing p-worlds can be represented using the aggregate `#count`. The expression

```
#count{ X :  P(X) } = k
```

is satisfied iff the set of t such that $P(t)$ holds has cardinality k. To express that a set $w(k)$ satisfies $(\mathrm{C}\to)$, we introduce the predicate `closedIMP`/1 defined by the following rule:

$$\begin{array}{rcl} \texttt{atSet}\,(w) & \text{iff} & w \subseteq \mathcal{A} \\ \texttt{belongsClosG}\,(A, w) & \text{iff} & A \in \mathrm{Cl}_G(w) \end{array}$$

```
closedIMP(W) :- atSet(W),
 #count{ At : member(At,W), def(At,imp(A1,A2)),
              belongsClosG(A1,W),
              not belongsClosG(A2,W) }  =  0.
```

Thus, $\texttt{closedIMP}(w)$ holds iff $w \subseteq \mathcal{A}$ and the set of at such that $at \in w$ and $at \coloneqq A_1 \to A_2$ and $A_1 \in \mathrm{Cl}_G(w)$ and $A_2 \notin \mathrm{Cl}_G(w)$ has cardinality 0; accordingly, w satisfies $(\mathrm{C}\to)$. The definition of predicates `closedNEG`/1 and `closedAT`/1, corresponding to the properties $(\mathrm{C}\neg)$ and (CAt) respectively, is similar. We can characterize p-worlds by introducing the predicate `pworld`/1 and the defining rule:

```
pworld(W) :-
    closedNEG(W), closedIMP(W), closedAT(W).
```

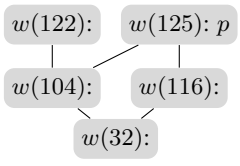
We describe how p-worlds are selected and turned into worlds of the countermodel; worlds are identified by the predicate `world`/1. Firstly, we have to choose the root of the countermodel, designated by the predicate `root`/1. To this aim, we introduce the following choice rule $(\mathrm{Ch_r})$:

```
{ root(W) : pworld(W),
    not belongsClosG(G,W) } = 1 :- goal(G).
```

The rule has a cardinality constraint in the head [2]: it forces the existence of exactly one root w_0 such that w_0 is a p-world and $G \notin \mathrm{Cl}_G(w_0)$, with G the encoding of the goal formula. We also need the defining rule '`world(W) :- root(W)`' stating that the root is a world (the first world inserted into the countermodel). Every chosen world must satisfy the saturation properties (S1) and (S2) and this might require the selection of new worlds. Let w be a p-world and $at \in \mathcal{A}$; w is a *self-witness* for at iff one of the following properties holds:

[1] Namely: $\psi^\star(a(k)) = \psi(a(k))$; $\psi^\star(\neg A) = \neg(\psi^\star(A))$; $\psi(A_1 \odot A_2) = \psi^\star(A_1) \odot \psi^\star(A_2)$ for $\odot \in \{\wedge, \vee, \to\}$.

[2] Cardinality constraints in the head do not require `#count`.



$w(125)$	$a(0)$, $a(2)$, $a(3)$, $a(4)$, $a(5)$, $a(6)$
$w(122)$	$a(1)$, $a(3)$, $a(4)$ $a(5)$, $a(6)$
$w(116)$	$a(2)$, $a(4)$, $a(5)$, $a(6)$
$w(104)$	$a(3)$, $a(5)$, $a(6)$
$w(32)$	$a(5)$

Figure 3: Countermodel for T (see Ex. 5)

(p1) $at \coloneqq \neg A_1$ and $A_1 \in \mathrm{Cl}_G(w)$;

(p2) $at \coloneqq A_1 \to A_2$ and $A_1 \in \mathrm{Cl}_G(w)$ and $A_2 \notin \mathrm{Cl}_G(w)$.

Such a relation can be easily encoded by the predicate `selfWitn`/2:

```
selfWitn(W, At) :- %(p1)
  def(At, neg(A1)), belongsClosG(A1,W).
selfWitn(W, At) :- def(At, imp(A1,A2)), %(p2)
  belongsClosG(A1,W), not belongsClosG(A2,W).
```

Let w be a world and $at \coloneqq \neg A_1$ or $at \coloneqq A_1 \to A_2$ such that $at \notin w$. A p-world w_1 is a *w-witness* for at, expressed by $\mathtt{witn}(w, at, w_1)$, iff $w \subset w_1$ and w_1 is a self-witness for at. If the world w is not a self-witness for at, we select a p-world w_1 such that $\mathtt{witn}(w, at, w_1)$ holds. Since we are interested in small models, we require that w_1 is unique, namely: if both $\mathtt{witn}(w, at, w_1)$ and $\mathtt{witn}(w, at, w_2)$ hold, then $w_1 = w_2$. Witnesses are selected by the following choice rule ($\mathrm{Ch_w}$):

$$\mathtt{le}(w_1, w_2) \quad \text{iff} \quad w_1 \subset w_2$$
$$\mathtt{negOrImp}(at) \quad \text{iff} \quad at \coloneqq \neg A_1 \text{ or } at \coloneqq A_1 \to A_2.$$

```
{ witn(W,At,W1) :
       pworld(W1), le(W,W1),
       selfWitn(W1, At) } = 1  :-
     world(W), negOrImp(At),
     not member(At,W), not selfWitn(W,At).
```

The selected witnesses are promoted to worlds by the rule '`world(W) :- witn(_,_,W)`'. If an answer set is found, the selected worlds constitute a complete G-saturated set, hence a countermodel for G. Finally, we instruct the solver to search for solutions minimizing the number of worlds, computed by the predicate `countWorlds`/1:

```
countWorlds(M) :- #count{ W : world(W) } = M.
#minimize { M : countWorlds(M) }.
```

To run the program Π_G:

```
clingo generator.lp goal.lp
```

where the files `generator.lp` and `goal.lp` encode the components `Gen` and `Goal`(G) respectively. To display the solutions, we introduce the following auxiliary predicates:

`countAtSets`/1:	number of subsets of $\mathrm{At}(G)$;
`countPW`/1:	number of p-worlds;
`succ`/2:	successor relation between worlds;
`forces`/2:	forcing relation on prop. variables.

Example 5 (Ex. 4 cont) We get the following answer set:

```
countAtSets(128) countPW(10) root(w(32))
countWorlds(5)  forces(w(125),p)
succ(w(32),w(104)) succ(w(32),w(116))
succ(w(104),w(122)) succ(w(104),w(125))
succ(w(116),w(125))       OPTIMUM FOUND
```

$D_4 \;=\; (p_1 \to p_2) \vee (p_2 \to p_1) \vee (q_1 \to q_2) \vee (q_2 \to q_1)$
$a(0) \mapsto p_1 \;\; a(1) \mapsto p_2 \;\; a(2) \mapsto q_1 \;\; a(3) \mapsto q_2 \;\; a(4) \mapsto p_1 \to p_2$
$a(5) \mapsto p_2 \to p_1 \quad a(6) \mapsto q_1 \to q_2 \quad a(7) \mapsto q_2 \to q_1$

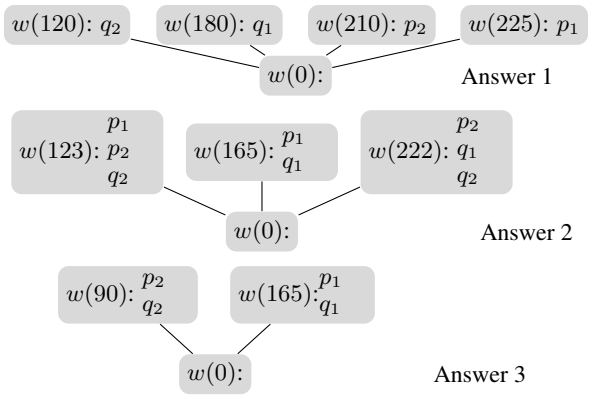
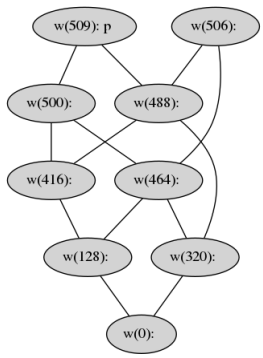


Figure 4: Countermodels for D_4 (see Ex. 6)

It corresponds to the minimal countermodel displayed in Fig. 3 (isomorphic to the one in Fig. 2). In the table we list the atoms contained in each world; the atom $a(6)$, encoding the goal formula T, does not belong to the root $w(32)$. ◇

Example 6 Let D_4 be the goal formula in Fig. 4, having 8 atomic subformulas. We have 49 p-worlds out of $2^8 = 256$ subsets of $\mathrm{At}(D_4)$. The generator computes three answer sets, corresponding to the models in Fig. 4. The first solution (Answer 1) has 5 worlds and it is not optimal. Then, the solver finds a solution with 4 worlds (Answer 2), finally an answer with 3 worlds (Answer 3), which is optimal. ◇

The previous example shows that in general a minimal countermodel for G cannot be obtained by shrinking a given countermodel for G. Indeed, the minimal countermodels for the formula D_4 in Fig. 4 are isomorphic to the bottom model: there is a root, not forcing any propositional variable, and two maximal worlds (w.r.t. $\leq$), each of them forcing exactly two propositional variables. The top model $\mathcal{K}_1$ in Fig. 4 has four maximal worlds, each of them forcing one variable. Using the standard filtration techniques [Chagrov and Zakharyaschev, 1997], there is no way to overlap or delete some of the maximal worlds in $\mathcal{K}_1$ and get a 3-world countermodel for D_4 (the solver had to discard the maximal worlds chosen to build $\mathcal{K}_1$ and to select new ones). Standard proof-search procedures, such as the ones mentioned in the Introduction, build countermodels isomorphic to or bigger than $\mathcal{K}_1$; indeed, to falsify D_4, at least four distinct maximal worlds are generated, one for each disjunct of D_4. Moreover, the computed maximal worlds have in general a "maximal forcing" (either exactly 3 variables or exactly 1 variable are forced), not matching the intermediate circumstance here required (exactly 2 variables are forced). We have tested two provers designed to reduce redundancies in proof-search, and both fail to build a minimal countermodel for D_4: STRIP [Larchey-Wendling *et al.*, 2001], based on structural sharing, yields a countermodel having 6 worlds (4 of which are maximal), the prover presented in [Fiorentini and Ferrari, 2017], implementing a for-



$N_1 = p \qquad N_2 = \neg p$
$N_{2n+3} = N_{2n+1} \vee N_{2n+2}$
$N_{2n+4} = N_{2n+3} \rightarrow N_{2n+1}$
$a(0) \mapsto p_1 \qquad a(1) \mapsto \neg p_1$
$a(k) :=$
$(\, a(0) \vee \cdots \vee a(k-1)\,)$
$\rightarrow (\, a(0) \vee \cdots \vee a(k-2)\,)$

goal: $a(0) \vee \cdots \vee a(8)$

Figure 5: Minimal countermodel for N_{17} (drawn with Graphviz)

ward proof-search strategy, outputs a 5-world countermodel isomorphic to $\mathcal{K}_1$.

Example 7 Let us consider the one-variable formulas N_i of the Nishimura family [Chagrov and Zakharyaschev, 1997] defined in Fig. 5, which are not valid in IPL. For N_{17}, having 9 atomic subformulas, we have $2^9 = 512$ subsets of $\mathrm{At}(N_{17})$ and only 10 p-worlds. The first answer corresponds to the countermodel in Fig. 5, which is isomorphic to the standard "tower-like" minimal countermodel for N_{17}. $\diamond$

We have also performed some experiments on the non-valid formulas of Intuitionistic Logic Theorem Proving (ILTP) Library [Raths *et al.*, 2007]. For each tested formula G, Table 1 reports the number of atomic subformulas, the number of p-worlds, the optimum (size of the minimal countermodel) and the CPU time required to compute it [3]. In all the cases, the number of p-worlds is considerably smaller than the number of subsets of $\mathrm{At}(G)$: if we worked on all the subsets of $\mathrm{At}(G)$, instead of restricting ourselves to p-worlds, or we used a naive generation algorithm, the computation would not terminate even with these small formulas. For all the formulas, the minimal countermodel has 2 worlds and coincides with the first answer; with bigger instances, there is no answer within 60 secs. The algorithm is very efficient on Nishimura formulas (see Ex. 7); this depends on the fact that the number of p-worlds essentially coincides with the size of the minimal countermodels.

4 Conclusion

We have presented an ASP program to generate minimal countermodels for non-valid formulas in IPL. Our approach is inspired by [Goré and Thomson, 2012; Goré *et al.*, 2014] with significant differences. Indeed, Goré&al. aim at defining an efficient procedure to test the validity of a goal formula G in IPL (which is a PSPACE-complete problem). To this aim, they build a model $\mathcal{K}_G$ such that G is valid iff $\mathcal{K}_G$ is not a countermodel for G. Worlds of $\mathcal{K}_G$ are represented by proper sets of atomic subformulas of G and, to get an efficient implementation, they exploit BDDs (Binary Decision Diagrams). Here, we deal with a harder problem, since we focus on minimal countermodels for G (and $\mathcal{K}_G$ in general is not small).

[3] Tests were conducted on a standard machine with a 3.0GHz Intel Core(TM)2 Duo CPU and 3.5GB memory,

Goal formula G	Subsets of $\mathrm{At}(G)$	p-worlds	Opt.	CPU time (in sec.)
SYJ207+1.002	32768	1280	2	11.85
SYJ208+2.002	8192	694	2	6.85
SYJ209+1.005	32768	921	2	11.50
SYJ210+1.004	65536	776	2	14.95
SYJ211+1.001	32768	534	2	7.21
SYJ212+1.005	65536	975	2	15.32

Table 1: Some tests on ILTP Library [Raths *et al.*, 2007]

Accordingly, we use a different encoding (for instance, $\neg$ is a primitive connective) and a different generation algorithms (we start from a p-world falsifying G and we saturate it by adding the required witnesses).

The program Π_G is quite general and modular; indeed, it can be immediately extended to deal with propositional super-intuitionistic logics based on Kripke semantics, provided that the frame conditions can be captured by the ASP language. An example is the Gödel-Dummett logic GD, obtained by adding to IPL the axiom schema $(A \rightarrow B) \vee (B \rightarrow A)$ and semantically characterized by linear frames. We can build countermodels in GD by adding to Π_G the following constraint (lin):

```
:-  world(W1), world(W2),  W1 <> W2,
    not le(W1,W2), not le(W2,W1).
```

This rule forbids the existence of two worlds w_1 and w_2 such that $w_1 \neq w_2$ and $w_1 \not\subset w_2$ and $w_2 \not\subset w_1$. Accordingly, $\Pi_G \cup \{(\mathrm{lin})\}$ is an ASP program to generate minimal countermodels in the logic GD. Other examples of intermediate logics that can be covered by extensions of Π_G are: the logic of bounded depth, the logic of bounded branch, the Here and There logic, closely connected with ASP [Pearce, 1997]. The crucial point is that frame conditions can be freely composed; for instance, if $(\mathrm{bd3})$ encodes the condition "the model has depth at most 3", the program $\Pi_G \cup \{(\mathrm{lin}), (\mathrm{bd3})\}$ computes linear countermodels for G having depth at most 3. We remark that standard provers do not enjoy this modularity; in general it is not even obvious how to extend a prover for IPL so as to cover the mentioned logics.

We defer to future work an in-depth classification of the frame conditions that can be expressed in ASP. We also plan to investigate (multi)-modal and temporal logics, also considering other notions of minimality, such as the ones discussed in [Papacchini and Schmidt, 2011; 2014].

Acknowledgments

I am grateful to the reviewers for their valuable suggestions. This work has been partially funded by the INdAM-GNCS project 2018 "Metodi di prova orientati al ragionamento automatico per logiche non-classiche".

References

[Baral, 2010] Chitta Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, 2010.

[Bry and Yahya, 2000] François Bry and Adnan H. Yahya. Positive unit hyperresolution tableaux and their applica-

tion to minimal model generation. *J. Autom. Reasoning*, 25(1):35–82, 2000.

[Chagrov and Zakharyaschev, 1997] Alexander V. Chagrov and Michael Zakharyaschev, *Modal Logic*. Oxford University Press, 1997.

[Cheney *et al.*, 2016] James Cheney, Alberto Momigliano, and Matteo Pessina. Advances in Property-Based Testing for αProlog. In *TAP*, volume 9762 of *LNCS*, pages 37–56. Springer, 2016.

[Claessen and Rosén, 2015] Koen Claessen and Dan Rosén. SAT modulo intuitionistic implications. In M. Davis et al., editors., *LPAR-20, Proceedings*, volume 9450 of *LNCS*, pages 622–637. Springer, 2015.

[Corsi and Tassi, 2007] Giovanna Corsi and Gabriele Tassi. Intuitionistic logic freed of all metarules. *J. Symb. Log.*, 72(4):1204–1218, 2007.

[Dantsin *et al.*, 2001] Evgeny Dantsin, Thomas Eiter, Georg Gottlob, and Andrei Voronkov. Complexity and expressive power of logic programming. *ACM Comput. Surv.*, 33(3):374–425, 2001.

[Demri *et al.*, 2014] Stéphane Demri, Deepak Kapur, and Christoph Weidenbach, editors. *IJCAR 2014, Proceedings*, volume 8562 of *LNCS*. Springer, 2014.

[Dummett, 1959] Michael Dummett. A propositional calculus with denumerable matrix. *J. Symbolic Logic*, 24(2):97–106, 1959.

[Dyckhoff, 2016] Roy Dyckhoff. Intuitionistic decision procedures since Gentzen. In *Advances in proof theory*, volume 28 of *Progr. Comput. Sci. Appl. Logic*, pages 245–267. Birkhäuser/Springer, [Cham], 2016.

[Ferrari *et al.*, 2013] Mauro Ferrari, Camillo Fiorentini, and Guido Fiorino. Contraction-free linear depth sequent calculi for intuitionistic propositional logic with the subformula property and minimal depth counter-models. *J. Autom. Reasoning*, 51(2):129–149, 2013.

[Ferrari *et al.*, 2015] Mauro Ferrari, Camillo Fiorentini, and Guido Fiorino. An evaluation-driven decision procedure for G3i. *ACM Trans. Comput. Log.*, 16(1):8:1–8:37, 2015.

[Fiorentini and Ferrari, 2017] Camillo Fiorentini and Mauro Ferrari. A forward unprovability calculus for intuitionistic propositional logic. In R. A. Schmidt and C. Nalon, editors, *TABLEAUX 2017*, volume 10501 of *LNCS*, pages 114–130. Springer, 2017.

[Galmiche and Larchey-Wendling, 1999] Didier Galmiche and Dominique Larchey-Wendling. Structural sharing and efficient proof-search in propositional intuitionistic logic. In P. S. Thiagarajan et al., editors, *Advances in Computing Science - ASIAN'99, Proceedings*, volume 1742 of *LNCS*, pages 101–112. Springer, 1999.

[Gebser *et al.*, 2011] Martin Gebser, Benjamin Kaufmann, Roland Kaminski, Max Ostrowski, Torsten Schaub, and Marius Thomas Schneider. Potassco: The Potsdam Answer Set Solving Collection. *AI Commun.*, 24(2):107–124, 2011.

[Gebser *et al.*, 2012] Martin Gebser, Roland Kaminsk, Benjamin Kaufmann, and Torsten Schaub. *Answer Set Solving in Practice*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan and Claypool Publishers, 2012.

[Goré and Thomson, 2012] Rajeev Goré and Jimmy Thomson. Bdd-based automated reasoning for propositional bi-intuitionistic tense logics. In B. Gramlich et al., editors, *IJCAR 2012, Proceedings*, volume 7364 of *LNCS*, pages 301–315. Springer, 2012.

[Goré *et al.*, 2014] Rajeev Goré, Kerry Olesen, and Jimmy Thomson. Implementing tableau calculi using bdds: Bddtab system description. In Demri et al. [2014], pages 337–343.

[Larchey-Wendling *et al.*, 2001] Dominique Larchey-Wendling, Dominique Méry, and Didier Galmiche. STRIP: Structural Sharing for Efficient Proof-Search. In R. Goré et al., editors, *IJCAR 2001, Proceedings*, volume 2083 of *LNCS*, pages 696–700. Springer, 2001.

[Lorenz, 1994] Sven Lorenz. A tableaux prover for domain minimization. *J. Autom. Reasoning*, 13(3):375–390, 1994.

[Negri, 2014] Sara Negri. Proofs and countermodels in non-classical logics. *Logica Universalis*, 8(1):25–60, 2014.

[Niemelä, 1996] Ilkka Niemelä. A tableau calculus for minimal model reasoning. In P. Miglioli et al., editors, *TABLEAUX '96, Proceedings*, volume 1071 of *LNCS*, pages 278–294. Springer, 1996.

[Papacchini and Schmidt, 2011] Fabio Papacchini and Renate A. Schmidt. A tableau calculus for minimal modal model generation. *Electr. Notes Theor. Comput. Sci.*, 278:159–172, 2011.

[Papacchini and Schmidt, 2014] Fabio Papacchini and Renate A. Schmidt. Terminating minimal model generation procedures for propositional modal logics. In Demri et al. [2014], pages 381–395.

[Pearce, 1997] David Pearce. A new logical characterisation of stable models and answer sets. In J. Dix et al., editors, *Non-Monotonic Extensions of Logic Programming*, pages 57–70, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.

[Pinto and Dyckhoff, 1995] Luis Pinto and Roy Dyckhoff. Loop-free construction of counter-models for intuitionistic propositional logic. In Behara et al., editor, *Symposia Gaussiana, Conference A*, pages 225–232. Walter de Gruyter, Berlin, 1995.

[Raths *et al.*, 2007] Thomas Raths, Jens Otten and Christoph Kreitz. The ILTP problem library for intuitionistic logic. *J. Autom. Reasoning*, 31:261–271, 2007.

[Skura, 2011] Tomasz Skura. Refutation Systems in Propositional Logic. In D. Gabbay et al., editors, *Handbook of Philosophical Logic*, volume 16, pages 115–157. Springer, Dordrecht, 2011.

[Svejdar, 2006] Vítezslav Svejdar. On sequent calculi for intuitionistic propositional logic. *Comment. Math. Univ. Carolinae*, 47(1):159–173, 2006.