SAT-based Proof Search in Intermediate Propositional Logics

Camillo Fiorentini[1][0000−0003−2152−7488] and Mauro Ferrari[2][0000−0002−7904−1125]

[1] Dep. of Computer Science, Università degli studi di Milano, Italy
[2] Dep. of Theoretical and Applied Sciences, Università degli studi dell'Insubria, Italy

Abstract. We present a decision procedure for intermediate logics relying on a modular extension of the SAT-based prover `intuitR` for IPL (Intuitionistic Propositional Logic). Given an intermediate logic L and a formula α, the procedure outputs either a Kripke countermodel for α or the instances of the characteristic axioms of L that must be added to IPL in order to prove α. The procedure exploits an incremental SAT-solver; during the computation, new clauses are learned and added to the solver.

1 Introduction

Recently, Claessen and Rosén have introduced `intuit` [4], an efficient decision procedure for Intuitionistic Propositional Logic (IPL) based on the Satisfiability Modulo Theories (SMT) approach. The prover language consists of (flat) clauses of the form $\bigwedge A_1 \to \bigvee A_2$ (with A_i a set of atoms), which are fed to the SAT-solver, and implication clauses of the form $(a \to b) \to c$ (a, b, c atoms); thus, we need an auxiliary clausification procedure to preprocess the input formula. The search is performed via a proper variant of the DPLL($\mathcal{T}$) procedure [16], by exploiting an incremental SAT-solver; during the computation, whenever a semantic conflict is thrown, a new clause is learned and added to the SAT-solver. As discussed in [9], there is a close connection between the `intuit` approach and the known proof-theoretic methods. Actually, the decision procedure mimics the standard root-first proof search strategy for a sequent calculus strongly connected with Dyckhoff's calculus LJT [5] (alias G4ip). To improve performances, we have re-designed the prover by adding a restart operation, thus obtaining `intuitR` [8] (`intuit` with Restart). Differently from `intuit`, the `intuitR` procedure has a simple structure, consisting of two nested loops. Given a formula α, if α is provable in IPL the call `intuitR(`α`)` yields a derivation of α in the sequent calculus introduced in [8], a plain calculus where derivations have a single branch. If α is not provable in IPL, the outcome of `intuitR(`α`)` is a (typically small) countermodel for α, namely a Kripke model falsifying α. We stress that `intuitR` is highly performant: on the basis of a standard benchmarks suite, it outperforms `intuit` and other state-of-the-art provers (in particular, `fCube` [6] and `intHistGC` [12]).

In this paper we present `intuitRIL`, an extension of `intuitR` to Intermediate Logics, namely propositional logics extending IPL and contained in CPL

(Classical Propositional Logic). Specifically, let α be a formula and L an axiomatizable intermediate logic having Kripke semantics; the call `intuitRIL(`α`,`L`)` tries to prove the validity of α in L. To this aim, the prover searches for a set Ψ containing instances of $\mathrm{Ax}(L)$, the characteristic axioms of L, such that α can be proved in IPL from Ψ. Note that this is different from other approaches, where the focus is on the synthesis of specific inference rules for the logic at hand (see, e.g., [17]). Basically, `intuitRIL(`α`,`L`)` searches for a countermodel $\mathcal{K}$ for α, exploiting the search engine of `intuitR`: whenever we get $\mathcal{K}$, we check whether $\mathcal{K}$ is a model of L. If this is the case, we conclude that α is not valid in L (and $\mathcal{K}$ is a witness to this). Otherwise, the prover selects an instance ψ of $\mathrm{Ax}(L)$ falsified in $\mathcal{K}$ (there exists at least one); ψ is acknowledged as learned axiom and, after clausification, it is fed to the SAT-solver. We stress that a naive implementation of the procedure, where at each iteration of the main loop the computation restarts from scratch, would be highly inefficient: each time the SAT-solver should be initialized by inserting all the clauses encoding the input problem and all the clauses learned so far. Instead, we exploit an incremental SAT-solver, where clauses can be added but never deleted (hence, all the simplifications and optimisations performed by the solver are preserved); note that this prevents us from exploiting strategies based on standard sequent/tableaux calculi, where backtracking is required.

If the call `intuitRIL(`α`,`L`)` succeeds, by tracking the computation we get a derivation $\mathcal{D}$ of α in the sequent calculus C_L (see Fig. 1); from $\mathcal{D}$ we can extract all the axioms learned during the computation. We stress that the procedure is quite modular: to handle a logic L, one has only to implement a specific learning mechanism for L (namely: if $\mathcal{K}$ is not a model of L, pick an instance of $\mathrm{Ax}(L)$ falsified in $\mathcal{K}$). The main drawback is that there is no general way to bound the learned axioms, thus termination must be investigated on a case-by-case basis. We guarantee termination for some relevant intermediate logics, such as Gödel-Dummett Logic GL, the family GL_n ($n \geq 1$) of Gödel-Dummett Logics with depth bounded by n (GL_1 coincides with Here and There Logic, well known for its applications in Answer Set Programming [15]) and Jankov Logic (for a presentation of such logics see [2]). As a corollary, for each of the mentioned logic L we get a bounding function [3], namely: given α, we compute a bounded set Ψ_α of instances of $\mathrm{Ax}(L)$ such that α is valid in L iff α is provable in IPL from assumptions Ψ_α; in general we improve the bounds in [1,3]. The `intuitRIL` Haskell implementation and other additional material (e.g., the omitted proofs) can be downloaded at `https://github.com/cfiorentini/intuitRIL`.

2 Basic definitions

Formulas, denoted by lowercase Greek letters, are built from an enumerable set of propositional variables $\mathcal{V}$, the constant $\bot$ and the connectives $\wedge$, $\vee$, $\rightarrow$; moreover, $\neg\alpha$ stands for $\alpha \rightarrow \bot$ and $\alpha \leftrightarrow \beta$ stands for $(\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$. Elements of the set $\mathcal{V} \cup \{\bot\}$ are called *atoms* and are denoted by lowercase Roman letters, uppercase Greek letters denote sets of formulas. By $\mathcal{V}_\alpha$ we denote the set of propositional variables occurring in α. The notation is extended to sets: $\mathcal{V}_\Gamma$ is the union of $\mathcal{V}_\alpha$ such that $\alpha \in \Gamma$; $\mathcal{V}_{\Gamma,\Gamma'}$ and $\mathcal{V}_{\Gamma,\alpha}$ stand for $\mathcal{V}_{\Gamma\cup\Gamma'}$ and $\mathcal{V}_{\Gamma\cup\{\alpha\}}$ respectively. A *substitution* is a map from propositional variables to formulas. By $[p_1 \mapsto \alpha_1, \ldots, p_n \mapsto \alpha_n]$ we denote the substitution χ such that $\chi(p) = \alpha_i$ if $p = p_i$ and $\chi(p) = p$ otherwise; the set $\{p_1, \ldots, p_n\}$ is the *domain* of χ, denoted by $\mathrm{Dom}(\chi)$; ϵ is the substitution having empty domain. The application of χ to a formula α, denoted by $\chi(\alpha)$, is defined as usual; $\chi(\Gamma)$ is the set of $\chi(\alpha)$ such that $\alpha \in \Gamma$. The *composition* $\chi_1 \cdot \chi_2$ is the substitution mapping p to $\chi_1(\chi_2(p))$.

A *(classical) interpretation* M is a subset of $\mathcal{V}$, identifying the propositional variables assigned to true. By $M \models \alpha$ we mean that α is true in M; $M \models \Gamma$ iff $M \models \alpha$ for every $\alpha \in \Gamma$. Classical Propositional Logic (CPL) is the set of formulas true in every interpretation. We write $\Gamma \vdash_{\mathrm{c}} \alpha$ iff $M \models \Gamma$ implies $M \models \alpha$, for every M. Note that α is CPL-valid (namely, $\alpha \in \mathrm{CPL}$) iff $\emptyset \vdash_{\mathrm{c}} \alpha$.

A (rooted) Kripke model is a quadruple $\langle W, \leq, r, \vartheta\rangle$ where W is a finite and non-empty set (the set of *worlds*), $\leq$ is a reflexive and transitive binary relation over W, the world r (the *root* of $\mathcal{K}$) is the minimum of W w.r.t. $\leq$, and $\vartheta : W \mapsto 2^{\mathcal{V}}$ (the *valuation* function) is a map obeying the persistence condition: for every pair of worlds w_1 and w_2 of $\mathcal{K}$, $w_1 \leq w_2$ implies $\vartheta(w_1) \subseteq \vartheta(w_2)$; the triple $\langle W, \leq, r\rangle$ is called *(Kripke) frame*. The valuation ϑ is extended to a *forcing* relation between worlds and formulas as follows:

$$w \Vdash p \text{ iff } p \in \vartheta(w), \forall p \in \mathcal{V} \qquad w \nVdash \bot \qquad w \Vdash \alpha \wedge \beta \text{ iff } w \Vdash \alpha \text{ and } w \Vdash \beta$$
$$w \Vdash \alpha \vee \beta \text{ iff } w \Vdash \alpha \text{ or } w \Vdash \beta \qquad w \Vdash \alpha \to \beta \text{ iff } \forall w' \geq w,\ w' \Vdash \alpha \text{ implies } w' \Vdash \beta.$$

By $w \Vdash \Gamma$ we mean that $w \Vdash \alpha$ for every $\alpha \in \Gamma$. A formula α is *valid* in the frame $\langle W, \leq, r\rangle$ iff for every valuation ϑ, $r \Vdash \alpha$ in the model $\langle W, \leq, r, \vartheta\rangle$. Propositional Intuitionistic Logic (IPL) is the set of formulas valid in all frames. Accordingly, if there is a model $\mathcal{K}$ such that $r \nVdash \alpha$ (here and below r designates the root of $\mathcal{K}$), then α is not IPL-valid; we call $\mathcal{K}$ a *countermodel* for α. We write $\Gamma \vdash_{\mathrm{i}} \delta$ iff, for every model $\mathcal{K}$, $r \Vdash \Gamma$ implies $r \Vdash \delta$; thus, α is IPL-valid iff $\emptyset \vdash_{\mathrm{i}} \alpha$.

Let L be one of the logics IPL and CPL; then, L is closed under modus ponens ($\{\alpha, \alpha \to \beta\} \subseteq L$ implies $\beta \in L$) and under substitution (for every χ, $\alpha \in L$ implies $\chi(\alpha) \in L$). An *intermediate logic* is any set of formulas L such that $\mathrm{IPL} \subseteq L \subseteq \mathrm{CPL}$, L is closed under modus ponens and under substitution. A model $\mathcal{K}$ is an L-model iff $r \Vdash L$; if $r \nVdash \alpha$, we say that $\mathcal{K}$ is an *L-countermodel* for α. An intermediate logic L can be characterized by a set of CPL-valid formulas, called the *L-axioms* and denoted by $\mathrm{Ax}(L)$. An L-axiom ψ of $\mathrm{Ax}(L)$ must be understood as a schematic formula, representing all the formulas of the kind $\chi(\psi)$; we call $\chi(\psi)$ an *instance* of ψ. Formally, $\mathrm{IPL} + \mathrm{Ax}(L)$ is the intermediate logic collecting the formulas α such that $\Psi \vdash_{\mathrm{i}} \alpha$, where Ψ is a finite set of instances of L-axioms from $\mathrm{Ax}(L)$. A *bounding function* for L is a map that, given α, yields a finite set Ψ_α of instances of L-axioms such that $\Psi_\alpha \vdash_{\mathrm{i}} \alpha$. If L admits a computable bounding function, we can reduce L-validity to IPL-validity (see [3] for an in-depth discussion). Let $\mathcal{F}$ be a class of frames and let $\mathrm{Log}(\mathcal{F})$ be the set of formulas valid in all frames of $\mathcal{F}$; then, $\mathrm{Log}(\mathcal{F})$ is an intermediate logic. A logic L has *Kripke semantics* iff there exists a class of frames $\mathcal{F}$ such that $L = \mathrm{Log}(\mathcal{F})$; we also say that L is characterized by $\mathcal{F}$. Henceforth, when we mention a logic L, we leave understood that L is an axiomatizable intermediate logic having Kripke semantics.

*Example 1 (*GL*).* A well-known intermediate logic is Gödel-Dummett logic GL [2], characterized by the class of linear frames. An axiomatization of GL is obtained by adding the linearity axiom $\mathbf{lin} = (a \to b) \lor (b \to a)$ to IPL. Using the terminology of [3], GL is formula-axiomatizable: a bounding function for GL is obtained by mapping α to the set Ψ_α of instances of **lin** where a and b are replaced with subformulas of α. In [1] it is proved that it is sufficient to consider the subformulas of α of the kind $p \in \mathcal{V}_\alpha$, $\neg\beta$, $\beta_1 \to \beta_2$. In Lemma 4 we further improve this bound tacking as bounding function the following map:

$$\begin{aligned}\mathrm{Ax}_{\mathrm{GL}}(\alpha) \;=\; & \{\, (a \to b) \lor (b \to a) \mid a, b \in \mathcal{V}_\alpha \,\} \;\cup\; \{\, (a \to \neg a) \lor (\neg a \to a) \mid a \in \mathcal{V}_\alpha \,\} \\ & \cup \{\, (a \to (a \to b)) \lor ((a \to b) \to a)) \mid a, b \in \mathcal{V}_\alpha \,\}\end{aligned}$$

Thus, if $\mathcal{V}_\alpha = \{a\}$, the only instance of **lin** to consider is $(a \to \neg a) \lor (\neg a \to a)$, independently of the size of α (the other instances are IPL-valid and can be omitted). As pointed out in [3], GL is not variable-axiomatizable, namely: it is not sufficient to consider instances of **lin** obtained by replacing a and b with variables from $\mathcal{V}_\alpha$. As an example, let $\alpha = \neg a \lor \neg\neg a$; α is GL-valid, the only variable-replacement instance of **lin** is $\psi_\alpha = (a \to a) \lor (a \to a)$ and $\psi_\alpha \nvdash_{\mathrm{i}} \alpha$. $\Diamond$

We review the main concepts about the clausification procedure described in [4]. *Clauses* φ and *implication clauses* λ are defined as

$$\begin{array}{lll} \varphi \;:=\; \bigwedge A_1 \to \bigvee A_2 \mid \bigvee A_2 & \qquad & \emptyset \subset A_k \subseteq \mathcal{V} \cup \{\bot\}, \text{ for } k \in \{1,2\} \\ \lambda \;:=\; (a \to b) \to c & & a \in \mathcal{V},\ \{b, c\} \subseteq \mathcal{V} \cup \{\bot\} \end{array}$$

where $\bigwedge A_1$ and $\bigvee A_2$ denote the conjunction and the disjunction of the atoms in A_1 and A_2 respectively ($\bigwedge\{a\} = \bigvee\{a\} = a$). Henceforth, $\bigwedge\emptyset \to \bigvee A_2$ must be read as $\bigvee A_2$; R, R_1, ... denote sets of clauses, X, X_1, ... sets of implication clauses. Given a set of implication clauses X, the *closure* of X, denoted by $(X)^\star$, is the set of clauses $b \to c$ such that $(a \to b) \to c \in X$.

The following lemma states some properties of clauses and closures.

Lemma 1. *(i)* $R \vdash_{\mathrm{i}} g$ *iff* $R \vdash_{\mathrm{c}} g$*, for every set of clauses* R *and every atom* g*.*
(ii) $X \vdash_{\mathrm{i}} b \to c$*, for every* $b \to c \in (X)^\star$*.*
(iii) $\Gamma \vdash_{\mathrm{i}} \alpha$ *iff* $\alpha \leftrightarrow g, \Gamma \vdash_{\mathrm{i}} g$*, where* $g \notin \mathcal{V}_{\Gamma,\alpha}$*.*

Clausification We assume a procedure `Clausify` that, given a formula α, computes sets of clauses R and X equivalent to α w.r.t. IPL. Formally, let α be a formula and let V be a set of propositional variables such that $\mathcal{V}_\alpha \subseteq V$. The procedure `Clausify(`$\alpha$`,`$V$`)` computes a triple (R, X, χ) satisfying:

(C1) $\Gamma, \alpha \vdash_{\mathrm{i}} \delta$ iff $\Gamma, R, X \vdash_{\mathrm{i}} \delta$, for every Γ and δ such that $\mathcal{V}_{\Gamma,\delta} \subseteq V$.
(C2) $\mathrm{Dom}(\chi) = \mathcal{V}_{R,X} \setminus V$ and $\mathcal{V}_{\chi(p)} \subseteq V$ for every $p \in \mathrm{Dom}(\chi)$.
(C3) $R, X \vdash_{\mathrm{i}} p \leftrightarrow \chi(p)$ for every $p \in \mathrm{Dom}(\chi)$.

$$\frac{R \vdash_{\mathrm{c}} g}{R, X \Rightarrow g}\,\mathrm{cpl}_0 \qquad \frac{R,\, A \vdash_{\mathrm{c}} b \qquad R, \varphi, X \Rightarrow g}{R, X \Rightarrow g}\,\mathrm{cpl}_1(\lambda) \qquad \begin{array}{l} \lambda = (a \to b) \to c \in X \\ A \subseteq \mathcal{V}_{R,X,g} \\ \varphi \;=\; \bigwedge(A \setminus \{a\}) \to c \end{array}$$

$$\frac{R, (X)^{\star}, X \Rightarrow g}{\Rightarrow \alpha}\,\mathrm{Claus}_0(g, \chi) \qquad \begin{array}{l} g \notin \mathcal{V}_{\alpha} \\ (R, X, \chi) = \texttt{Clausify(}\alpha \leftrightarrow g,\ \mathcal{V}_{\alpha,g}\texttt{)} \end{array}$$

$$\frac{R, R', (X')^{\star}, X, X' \Rightarrow g}{R, X \Rightarrow g}\,\mathrm{Claus}_1(\psi, \chi) \qquad \begin{array}{l} \psi \in \mathrm{Ax}(L,\, \mathcal{V}_{R,X,g}) \\ (R', X', \chi) = \texttt{Clausify(}\psi,\ \mathcal{V}_{R,X,g}\texttt{)} \end{array}$$

R is a set of clauses
X is a set of implication clauses
g is an atom

$$\pi(\rho) \;=\; \begin{cases} \langle\, \emptyset,\ [g \mapsto \alpha] \cdot \chi \,\rangle & \text{if } \rho = \mathrm{Claus}_0(g, \chi) \\ \langle\, \{\psi\},\ \chi \,\rangle & \text{if } \rho = \mathrm{Claus}_1(\psi, \chi) \\ \langle\, \emptyset,\ \epsilon \,\rangle & \text{otherwise} \end{cases}$$

Fig. 1. The sequent calculus C_L.

Basically, clausification introduces new propositional variables to represent subformulas of α; as a result we obtain a substitution χ which tracks the mapping on the new variables. Condition (C1) states that α can be replaced by $R \cup X$ in IPL reasoning. By (C2) the domain of χ consists of the new variables introduced in the clausification process. The following properties easily follow by (C1)-(C3):

(P1) $R, X \vdash_{\mathrm{i}} \alpha$. (P2) $R, X \vdash_{\mathrm{i}} \beta \leftrightarrow \chi(\beta)$ for every formula β.

We exploit a `Clausify` procedure essentially similar to the one described in [4], with slight modifications in order to match (C3). As discussed in [4], in IPL we can use a weaker condition (either $R, X \vdash_{\mathrm{i}} p \to \chi(p)$ or $R, X \vdash_{\mathrm{i}} \chi(p) \to p$ according to the case). It is not obvious whether the weaker condition should be more efficient; in many cases strong equivalences are more performant, maybe because they trigger more simplifications in the SAT-solver.

Example 2. Let $\alpha = (a \to b) \vee (b \to a)$ and $V = \{a, b\}$. The call `Clausify(`α`,`V`)` introduces the new variables $\tilde{p}_0$ and $\tilde{p}_1$ associated with the subformulas $a \to b$ and $b \to a$ respectively. Accordingly, the obtained sets R and X must satisfy $R, X \vdash_{\mathrm{i}} \tilde{p}_0 \leftrightarrow (a \to b)$ and $R, X \vdash_{\mathrm{i}} \tilde{p}_1 \leftrightarrow (b \to a)$. We get:

$$\begin{array}{ll} R \;= \{\, \tilde{p}_0 \vee \tilde{p}_1,\ \tilde{p}_0 \wedge a \to b,\ \tilde{p}_1 \wedge b \to a \,\} & \qquad \chi \;=\; [\, \tilde{p}_0 \mapsto a \to b,\ \tilde{p}_1 \mapsto b \to a \,] \\ X \;= \{\, (a \to b) \to \tilde{p}_0,\ (b \to a) \to \tilde{p}_1 \,\} & \end{array}$$

$\Diamond$

3 The Calculus C_L

Let L be an intermediate logic; we introduce the sequent calculus C_L to prove L-validity. We assume that L is axiomatized by a set $\mathrm{Ax}(L)$ of L-axioms; by

$$\cfrac{\cdots \qquad \cfrac{R_{n-1} \vdash_{\mathrm{c}} g}{R_{n-1}, X_{n-1} \Rightarrow g}\,\rho_n = \mathrm{cpl}_0}{R_{n-2}, X_{n-2} \Rightarrow g}\,\rho_{n-1}$$

$$\vdots$$

$$\cfrac{\cfrac{\cdots \qquad R_1, X_1 \Rightarrow g}{R_0, X_0 \Rightarrow g}\,\rho_1}{\Rightarrow \alpha}\,\rho_0 = \mathrm{Claus}_0$$

$$\forall i \in \{1, \ldots, n-1\},\ \rho_i = \mathrm{cpl}_1 \text{ or } \rho_i = \mathrm{Claus}_1$$

$$\pi(\mathcal{D}) = \langle\, \Psi_0 \cup \cdots \cup \Psi_n \,,\ \chi_0 \cdot \ldots \cdot \chi_n \,\rangle$$

$$\text{where } \langle \Psi_j, \chi_j \rangle = \pi(\rho_j)$$

Fig. 2. A C_L-derivation of $\Rightarrow \alpha$.

$\mathrm{Ax}(L, V)$ we denote the set of instances ψ of L-axioms such that $\mathcal{V}_\psi \subseteq V$. The calculus relies on a clausification procedure `Clausify` satisfying conditions (C1)–(C3) and acts on sequents $\Gamma \Rightarrow \delta$ such that:

- either $\Gamma = \emptyset$ or $\Gamma = R \cup X$ and $(X)^\star \subseteq R$ and δ is an atom.

Rules of C_L are displayed in Fig. 1. Rule cpl_0 (initial rule) can only be applied if the condition $R \vdash_{\mathrm{c}} g$ holds; if this is the case, the conclusion $R, X \Rightarrow g$ is an initial sequent, namely a top sequent of a derivation. The other rules depend on parameters that are made explicit in the rule name. A bottom-up application of cpl_1 requires the choice of an implication clause $\lambda = (a \to b) \to c$ from X, we call the *main formula*, and the selection of a set of atoms $A \subseteq \mathcal{V}_{R,X,g}$ such that $R, A \vdash_{\mathrm{c}} b$, where b is the middle variable in λ. As discussed in [8,9], cpl_1 is a sort of generalization of the rule $L \to\to$ of the sequent calculus LJT/G4ip for IPL [5,18]. Rules Claus_0 and Claus_1 exploit the clausification procedure. Rule Claus_0 requires the clausification of the formula $\alpha \leftrightarrow g$, with g a new atom ($g \notin \mathcal{V}_\alpha$); in rule Claus_1, the clausified formula ψ is selected from $\mathrm{Ax}(L, \mathcal{V}_{R,X,g})$. In both cases, the clauses returned by `Clausify` are stored in the premise of the applied rule and the computed substitution χ is displayed in the rule name; moreover, Claus_0 is annotated with the new atom g and Claus_1 with the chosen L-axiom ψ. To recover the relevant information associated with the application of a rule ρ, in Fig. 1 we define the pair $\pi(\rho) = \langle \Psi, \chi \rangle$, where Ψ is a set of instances of L-axioms and χ is a substitution. C_L-trees and C_L-derivations are defined as usual (see e.g. [18]); a sequent σ is provable in C_L iff there exists a C_L-derivation having root sequent σ. Let us consider a C_L-derivation $\mathcal{D}$ of $\Rightarrow \alpha$ (see Fig. 2). Reading the derivation bottom-up, the first applied rule is Claus_0. After such an application, the obtained sequents have the form $\sigma_k = R_k, X_k \Rightarrow g$, where $R_k \cup X_k$ is non-empty, thus rule Claus_0 cannot be applied any more; the rule applied at the top is cpl_0. Note that $\mathcal{D}$ contains a unique branch, consisting of the sequents $\Rightarrow \alpha$, $\sigma_0, \ldots, \sigma_{n-1}$. In Fig. 2 we also define the pair $\pi(\mathcal{D}) = \langle \Psi, \chi \rangle$: Ψ collects the (instances of) L-axioms selected by rule Claus_1, χ is obtained by composing the substitutions associated with the applied rules. The definition of $\pi(\mathcal{T})$, with $\mathcal{T}$ a C_L-tree, is similar. By $\mathcal{T}(\alpha; R, X \Rightarrow g)$ we denote a C_L-tree having root $\Rightarrow \alpha$ and leaf $R, X \Rightarrow g$. Given a C_L-tree $\mathcal{T}$, $\mathcal{V}_\mathcal{T}$ is the set of variables occurring in $\mathcal{T}$. We state some properties about C_L-trees:

Lemma 2. *Let $\mathcal{T} = \mathcal{T}(\alpha; R, X \Rightarrow g)$ and let $\pi(\mathcal{T}) = \langle \Psi, \chi \rangle$.*

(i) $\mathcal{V}_{\chi(p)} \subseteq \mathcal{V}_\alpha$*, for every* $p \in \mathcal{V}_\mathcal{T}$*.*
(ii) $R, X \vdash_{\mathrm{i}} \beta \leftrightarrow \chi(\beta)$*, for every formula* β*.*
(iii) If $R, X, \Gamma \vdash_{\mathrm{i}} g$ *and* $\mathcal{V}_\Gamma \subseteq \mathcal{V}_\alpha$*, then* $\Gamma, \chi(\Psi) \vdash_{\mathrm{i}} \alpha$*.*

Proposition 1. *Let $\mathcal{D}$ be a C_L-derivation of $\Rightarrow \alpha$ and let $\pi(\mathcal{D}) = \langle \Psi, \chi \rangle$. Then, $\mathcal{V}_{\chi(\Psi)} \subseteq \mathcal{V}_\alpha$ and $\chi(\Psi) \vdash_{\mathrm{i}} \alpha$.*

Proof. Since $\mathcal{D}$ is a C_L-derivation, $\mathcal{D}$ has the form depicted on the right where $\mathcal{T} = \mathcal{T}(\alpha; R, X \Rightarrow g)$; note that $\pi(\mathcal{T}) = \pi(\mathcal{D}) = \langle \Psi, \chi \rangle$. Since $R \vdash_{\mathrm{c}} g$, by Lemma 1(i) we get $R \vdash_{\mathrm{i}} g$, hence $R, X \vdash_{\mathrm{i}} g$. We can apply Lemma 2 and claim that $\mathcal{V}_{\chi(\Psi)} \subseteq \mathcal{V}_\alpha$ and $\chi(\Psi) \vdash_{\mathrm{i}} \alpha$. □

$$\mathcal{D} \;=\; \begin{array}{c} \dfrac{R \vdash_{\mathrm{c}} g}{R, X \Rightarrow g}\,\mathrm{cpl}_0 \\ \vdots\ \mathcal{T} \\ \Rightarrow \alpha \end{array}$$

Given a C_L-derivation $\mathcal{D}$ of $\Rightarrow \alpha$, Prop. 1 exhibits how to extract a set of instances Ψ_α of the L-axioms such that $\Psi_\alpha \vdash_{\mathrm{i}} \alpha$. If $\mathcal{D}$ does not contain applications of rule Claus_1, Ψ_α is empty, and this ascertains that α is IPL-valid; actually, $\mathcal{D}$ can be immediately embedded into the calculus for IPL introduced in [8]. As an immediate consequence of Prop. 1, we get the soundness of C_L: if $\Rightarrow \alpha$ is provable in C_L, then α is L-valid.

Even though C_L-derivations have a simple structure, the design of a root-first proof search strategy for C_L is far from being trivial. After having applied rule Claus_0 to the root sequent $\Rightarrow \alpha$, we enter a loop where at each iteration k we search for a derivation of $\sigma_k = R_k, X_k \Rightarrow g$. It is convenient to firstly check whether $R_k \vdash_{\mathrm{c}} g$ so that, by applying rule cpl_0, we immediately close the derivation at hand. To check classical provability, we exploit a SAT-solver; each time the solver is invoked, the set R_k has increased, thus it is advantageous to use an incremental SAT-solver. If $R_k \nvdash_{\mathrm{c}} g$, we have to apply either rule cpl_1 or rule Claus_1, but it is not obvious which strategy should be followed. First, we have to select one between the two rules. If rule cpl_1 is chosen, we have to guess proper λ and A; otherwise, we have to apply Claus_1, and this requires the selection of an instance ψ of an L-axiom. In any case, if we followed a blind choice, the procedure would be highly inefficient. To guide proof search, we follow a different approach based on countermodel construction; to this aim, we introduce a representation of Kripke models where worlds are classical interpretations ordered by inclusion.

Countermodels Let W be a finite set of interpretations with minimum M_0, namely: $M_0 \subseteq M$ for every $M \in W$. By $\mathcal{K}(W)$ we denote the Kripke model $\langle W, \leq, M_0, \vartheta \rangle$ where $\leq$ coincides with the subset relation $\subseteq$ and ϑ is the identity map, thus $M \Vdash p$ (in $\mathcal{K}(W)$) iff $p \in M$. We introduce the following *realizability relation* $\triangleright_W$ between elements of W and implication clauses:

$$\begin{array}{lll} M \triangleright_W (a \to b) \to c & \text{iff} & (a \in M) \text{ or } (b \in M) \text{ or } (c \in M) \text{ or} \\ & & (\ \exists M' \in W \text{ s.t. } M \subset M' \text{ and } a \in M' \text{ and } b \notin M'\ ). \end{array}$$

By $M \triangleright_W X$ we mean that $M \triangleright_W \lambda$ for every $\lambda \in X$. We state the crucial properties of the model $\mathcal{K}(W)$:

Proposition 2. *Let $\mathcal{K}(W)$ be the model generated by W and let $w \in W$. Let φ be a clause and $\lambda = (a \to b) \to c$ an implication clause.*

(i) If $w' \models \varphi$, for every $w' \in W$ such that $w \leq w'$, then $w \Vdash \varphi$.
(ii) If $w' \models b \to c$ and $w' \rhd_W \lambda$, for every $w' \in W$ such that $w \leq w'$, then $w \Vdash \lambda$.

Let $\mathcal{K}(W)$ be a model with root r, and assume that every interpretation w in W is a model of R; our goal is to get $r \Vdash R \cup X$ (where $(X)^* \subseteq R$), possibly by filling W with new worlds. To this aim, we exploit Prop. 2. By our assumption and point (i), we claim that $r \Vdash R$. Suppose that there is $w \in W$ and $\lambda = (a \to b) \to c \in X$ such that $w \not\rhd_W \lambda$; is it possible to amend $\mathcal{K}(W)$ in order to match (ii) and conclude $r \Vdash X$? By definition of $\rhd_W$, none of the atoms a, b, c belongs to w; moreover $\mathcal{K}(W)$ lacks a world w' such that $w \subset w'$ and $a \in w'$ and $b \notin w'$. We can try to fix $\mathcal{K}(W)$ by inserting the missing world w'; to preserve (i), we also need $w' \models R$. Accordingly, such a w' exists if and only if $R, w, a \nvdash_{\mathrm{c}} b$. This can be checked by querying a SAT-solver; moreover, if $R, w, a \nvdash_{\mathrm{c}} b$, the solver also computes the required w'. This completion process must be iterated until $\mathcal{K}(W)$ has been saturated with all the missing worlds or we get stuck. It is easy to check that the process eventually terminates. This is one of the key ideas beyond the procedure `intuitRIL` we present in next section.

4 The Procedure `intuitRIL`

We present the procedure `intuitRIL` (`intuit` with Restart for Intermediate Logics) that, given a formula α and a logic $L = \mathrm{IPL} + \mathrm{Ax}(L)$, returns either a set of L-axioms Ψ_α or a model $\mathcal{K}(W)$ with the following properties:

(Q1) If `intuitRIL(`α`,`L`)` returns Ψ_α, then $\Psi_\alpha \subseteq \mathrm{Ax}(L, \mathcal{V}_\alpha)$ and $\Psi_\alpha \vdash_{\mathrm{i}} \alpha$.
(Q2) If `intuitRIL(`α`,`L`)` returns $\mathcal{K}(W)$, then $\mathcal{K}(W)$ is an L-countermodel for α.

Thus, α is L-valid in the former case, not L-valid in the latter. If `intuitRIL(`α`,`L`)` returns Ψ_α, by tracing the computation we can build a C_L-derivation $\mathcal{D}$ of $\Rightarrow \alpha$ such that $\Psi_\alpha = \chi(\Psi)$, where $\langle \Psi, \chi \rangle = \pi(\mathcal{D})$; this certificates that $\Psi_\alpha \vdash_{\mathrm{i}} \alpha$.

The procedure is described by the flowchart in Fig. 3 and exploits a single incremental SAT-solver s: clauses can be added to s but not removed; by $\mathrm{R}(s)$ we denote the set of clauses stored in s. The SAT-solver is required to support the following operations:

– `newSolver(`R`)` creates a new SAT-solver initialized with the clauses in R.
– `addClauses(`s`, `R`)` adds the clauses in R to the SAT-solver s.
– `satProve(`s`, `A`, `g`)` calls s to decide whether $\mathrm{R}(s), A \vdash_{\mathrm{c}} g$ (A is a set of propositional variables). The solver outputs one of the following answers:
  • Yes(A'): thus, $A' \subseteq A$ and $\mathrm{R}(s), A' \vdash_{\mathrm{c}} g$;
  • No(M): thus, $A \subseteq M \subseteq \mathcal{V}_{\mathrm{R}(s)} \cup A$ and $M \models \mathrm{R}(s)$ and $g \notin M$.
  In the former case it follows that $\mathrm{R}(s), A \vdash_{\mathrm{c}} g$, in the latter $\mathrm{R}(s), A \nvdash_{\mathrm{c}} g$.

The computation of `intuitRIL(`α`,`L`)` consists of the following steps:

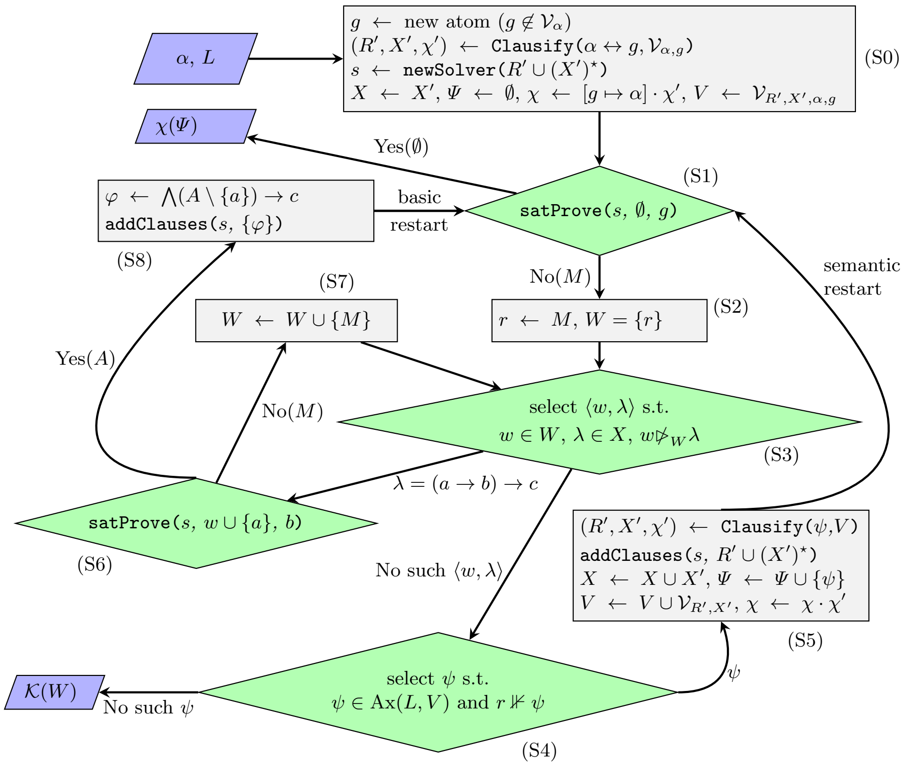

α, L
$g \leftarrow$ new atom ($g \notin \mathcal{V}_\alpha$)
$(R', X', \chi') \leftarrow$ `Clausify`$(\alpha \leftrightarrow g, \mathcal{V}_{\alpha,g})$
$s \leftarrow$ `newSolver`$(R' \cup (X')^\star)$
$X \leftarrow X', \Psi \leftarrow \emptyset, \chi \leftarrow [g \mapsto \alpha] \cdot \chi', V \leftarrow \mathcal{V}_{R',X',\alpha,g}$
(S0)
$\chi(\Psi)$
Yes($\emptyset$)
(S1)
`satProve`$(s, \emptyset, g)$
$\varphi \leftarrow \bigwedge(A \setminus \{a\}) \to c$
`addClauses`$(s, \{\varphi\})$
basic
restart
(S8)
No(M)
semantic
restart
(S7)
$W \leftarrow W \cup \{M\}$
$r \leftarrow M, W = \{r\}$
(S2)
Yes(A)
No(M)
select $\langle w, \lambda \rangle$ s.t.
$w \in W, \lambda \in X, w \not\triangleright_W \lambda$
(S3)
$\lambda = (a \to b) \to c$
`satProve`$(s, w \cup \{a\}, b)$
(S6)
$(R', X', \chi') \leftarrow$ `Clausify`(ψ, V)
`addClauses`$(s, R' \cup (X')^\star)$
$X \leftarrow X \cup X', \Psi \leftarrow \Psi \cup \{\psi\}$
$V \leftarrow V \cup \mathcal{V}_{R',X'}, \chi \leftarrow \chi \cdot \chi'$
No such $\langle w, \lambda \rangle$
(S5)
ψ
$\mathcal{K}(W)$
No such ψ
select ψ s.t.
$\psi \in \mathrm{Ax}(L, V)$ and $r \nVdash \psi$
(S4)

Fig. 3. Computation of `intuitRIL`(α, L).

(S0) The formula $\alpha \leftrightarrow g$, with g new propositional variable, is clausified. The outcome (R', X', χ') is used to create a new SAT-solver s and to properly initialize the global variables X (set of implication clauses), Ψ (set of L-axiom instances), V (set of propositional variables) and χ (substitution).

(S1) A loop starts *(main loop)*. The SAT-solver s is called to check whether $\mathrm{R}(s) \vdash_{\mathrm{c}} g$. If the answer is Yes($\emptyset$), the computation stops yielding $\chi(\Psi)$. Otherwise, the output is No(M) and the computation continues at Step (S2).

(S2) We set $r = M$ (the root of $\mathcal{K}(W)$) and $W = \{r\}$.

(S3) A loop starts *(inner loop)*. We have to select a pair $\langle w, \lambda \rangle$ such that $w \in W$, $\lambda \in X$ and $w \not\triangleright_W \lambda$. If such a pair does not exist, the inner loop ends and next step is (S4), otherwise the inner loop continues at Step (S6).

(S4) As we show in Lemma 3, at this point $\mathcal{K}(W)$ is a countermodel for α. If all the axioms in $\mathrm{Ax}(L, V)$ are forced at the root r of $\mathcal{K}(W)$, then $\mathcal{K}(W)$ is an L-countermodel for α and the computation ends returning $\mathcal{K}(W)$. Otherwise, we select ψ from $\mathrm{Ax}(L, V)$ such that $r \nVdash \psi$ and the computation continues at Step (S5); we call ψ the *learned axiom*.

(S5) We clausify ψ and we update the global variables. The computation restarts from Step (S1) with a new iteration of the main loop (*semantic restart*).

(S6) Let $\langle w, (a \to b) \to c\rangle$ be the pair selected at Step (S3). The SAT-solver s is called to check whether $\mathrm{R}(s), w, a \vdash_{\mathrm{c}} b$. If the result is $\mathrm{No}(M)$, the inner loop continues at step (S7). Otherwise, the answer is $\mathrm{Yes}(A)$; the inner loop ends and the computation continues at Step (S8).

(S7) The interpretation M is added to W and the computation continues at Step (S3) with a new iteration of the inner loop.

(S8) The clause φ (*learned basic clause*) is added to the SAT-solver s and the computation restarts from Step (S1) (*basic restart*).

Intuitively, `intuitRIL(`α,L`)` searches for an L-countermodel $\mathcal{K}(W)$ for α. In the construction of $\mathcal{K}(W)$, whenever a conflict arises, a restart operation is triggered. A basic restart happens when it is not possible to fill the set W with a missing world (see the discussion after Prop. 2). A semantic restart is thrown when $\mathcal{K}(W)$ is a countermodel for α but it fails to be an L-model. In either case, the construction of $\mathcal{K}(W)$ restarts from scratch. However, to prevent that the same kind of conflict shows up again, new clauses are learned and fed to the SAT-solver (this complies with DPLL($\mathcal{T}$) with learning computation paradigm [16]). If the outcome is $\chi(\Psi)$, by tracing the computation we can build a C_L-derivation $\mathcal{D}$ of $\Rightarrow \alpha$ such that $\pi(\mathcal{D}) = \langle \Psi, \chi\rangle$. The derivation is built bottom-up. The initial Step (S0) corresponds to the application of rule Claus_0 to the root sequent $\Rightarrow \alpha$; basic and semantic restarts bottom-up expand the derivation by applying rule cpl_1 and Claus_1 respectively. We stress that the procedure is quite modular; to treat a specific logic L one has only to provide a concrete implementation of Step (S4). For $L = \mathrm{IPL}$, Step (S4) is trivial, since the set $\mathrm{Ax}(\mathrm{IPL}, V)$ is empty. Actually, `intuitRIL` applied to IPL has the same behaviour as the procedure `intuitR` introduced in [8].

Example 3. Let us consider *Jankov axiom* **wem** $= \neg a \vee \neg\neg a$ [2,13] (aka *weak excluded middle*), which holds in all frames having a single maximal world (thus, **wem** is GL-valid). The trace of the execution of `intuitRIL(`**wem**,GL`)` is shown in Fig. 4. The initial clausification yields $(R_0, X_0, \tilde{g})$, where X_0 consists of the implication clauses λ_0, λ_1 in Fig. 4 and R_0 contains the 7 clauses below:

$$\tilde{g} \to \tilde{p}_2, \quad \tilde{p}_0 \to \tilde{p}_2, \quad a \wedge \tilde{p}_0 \to \bot, \quad \tilde{p}_1 \to \tilde{p}_2, \quad \tilde{p}_0 \wedge \tilde{p}_1 \to \bot, \quad \tilde{p}_2 \to \tilde{g}, \quad \tilde{p}_2 \to \tilde{p}_0 \vee \tilde{p}_1.$$

Each row in Fig. 4 displays the validity tests performed by the SAT-solver and the computed answers. If the result is $\mathrm{No}(M)$, the last two columns show the worlds w_k in the current set W and, for each w_k, the list of λ such that $w \not\triangleright_W \lambda$; the pair selected for the next step is underlined. For instance, after call (1) we have $W = \{w_0\}$, $w_0 \not\triangleright_W \lambda_0$ and $w_0 \not\triangleright_W \lambda_1$; the selected pair is $\langle w_0, \lambda_0\rangle$. After call (2), the set W is updated by adding the world w_1; we have $w_1 \triangleright_W \lambda_0$, $w_1 \triangleright_W \lambda_1$, $w_0 \triangleright_W \lambda_0$ and $w_0 \not\triangleright_W \lambda_1$. Whenever the SAT-solver outputs $\mathrm{Yes}(A)$, we display the learned clause ψ_k. The SAT-solver is invoked 18 times and there are 6 restarts (1 semantic, 5 basic). After (3), we get $W = \{w_0, w_1, w_2\}$ and no pair $\langle w, \lambda\rangle$ can be selected, hence the model $\mathcal{K}(W)$ (displayed in the figure) is a countermodel for **wem**. However, $\mathcal{K}(W)$ is not a GL-model (indeed, it is not linear), hence we choose an instance of the linearity axiom not forced at w_0, namely ψ_0, and we force a semantic restart. The clausification of ψ_0 produces 6 new clauses and the new implication clauses λ_2, λ_3, λ_4. After each restart, the sets R_j are:

$$\begin{aligned} R_1 &= R_0 \cup \{\, \tilde{p}_3 \to \tilde{p}_4,\; a \to \tilde{p}_5,\; \tilde{p}_3 \wedge \tilde{p}_5 \to a,\; a \wedge \tilde{p}_4 \to \tilde{p}_3,\; a \wedge \tilde{p}_3 \to \bot,\; \tilde{p}_4 \vee \tilde{p}_5 \,\} \\ R_j &= R_{j-1} \cup \{\psi_{j-1}\} \quad \text{for } 2 \le j \le 6 \text{ (the } \psi_j\text{'s are defined in Fig. 4).} \end{aligned}$$

The C_{GL}-derivation of $\Rightarrow \neg a \vee \neg\neg a$ extracted from the computation is:

$$\dfrac{\dfrac{\dfrac{R_1, a, \tilde{p}_0 \vdash_{\mathrm{c}} \bot \qquad \dfrac{R_2, a, \tilde{p}_0 \vdash_{\mathrm{c}} \bot \qquad \dfrac{R_3, a, \tilde{p}_3 \vdash_{\mathrm{c}} \bot \qquad \dfrac{R_4, \tilde{p}_0, \tilde{p}_5 \vdash_{\mathrm{c}} \bot \qquad \dfrac{R_5, a, \tilde{p}_4 \vdash_{\mathrm{c}} \bot \qquad \dfrac{R_6 \vdash_{\mathrm{c}} \tilde{g}}{R_6, X_1 \Rightarrow \tilde{g}}\,\mathrm{cpl}_0}{R_5, X_1 \Rightarrow \tilde{g}}\,\mathrm{cpl}_1(\lambda_1)}{R_4, X_1 \Rightarrow \tilde{g}}\,\mathrm{cpl}_1(\lambda_0)}{R_3, X_1 \Rightarrow \tilde{g}}\,\mathrm{cpl}_1(\lambda_1)}{R_2, X_1 \Rightarrow \tilde{g}}\,\mathrm{cpl}_1(\lambda_0)}{R_1, X_1 \Rightarrow \tilde{g}}\,\mathrm{cpl}_1(\lambda_3)}{R_0, X_0 \Rightarrow \tilde{g}}\,\mathrm{Claus}_1(\psi_0, \chi_1)}{\Rightarrow \neg a \vee \neg\neg a}\,\mathrm{Claus}_0(\tilde{g}, \chi_0)$$

◊

Now, we discuss partial correctness and termination of `intuitRIL`. Let us denote with $\sim_{\mathrm{c}}$ classical equivalence ($\alpha \sim_{\mathrm{c}} \beta$ iff $\vdash_{\mathrm{c}} \alpha \leftrightarrow \beta$) and with $\sim_{\mathrm{i}}$ intuitionistic equivalence ($\alpha \sim_{\mathrm{i}} \beta$ iff $\vdash_{\mathrm{i}} \alpha \leftrightarrow \beta$). We introduce some notation.

(†) The following terms refer to the configuration at the beginning of iteration k ($k \ge 0$), just after the execution of Step (S2):
- Φ_k is the set collecting all the learned basic clauses;
- R_k is the set of clauses stored in the SAT-solver s;
- X_k, Ψ_k, V_k, χ_k, r_k are the values of the corresponding global variables.

In Fig. 5 we inductively define the C_L-tree $\mathcal{T}_k$, having the form $\mathcal{T}(\alpha; R_k, X_k \Rightarrow g)$. In the application of rule Claus_0, g and χ' are defined as in Step (S0). In rule cpl_1, λ is the implication clause selected at iteration $k-1$ (of the main loop) in the last execution of Step (S3); A is the value computed at Step (S6) of iteration $k-1$. In the application of rule Claus_1, ψ and χ' are defined as in the execution of Step (S4) and (S5) of iteration $k-1$. One can easily check that the applications of the rules are sound. If Step (S1) yields $\mathrm{Yes}(\emptyset)$, we can turn $\mathcal{T}_k$ into a C_L-derivation by applying rule cpl_0.

Next lemma states some relevant properties of the computations of `intuitRIL`.

Lemma 3. *Let us consider the execution of iteration k of the main loop ($k \ge 0$).*

(i) $(X_k)^\star \cup \Phi_k \subseteq R_k$.
(ii) $V_k = \mathcal{V}_{\mathcal{T}_k}$ *and* $\Psi_k \subseteq \mathrm{Ax}(L, V_k)$ *and* $\pi(\mathcal{T}_k) = \langle \Psi_k, \chi_k \rangle$.
(iii) $\mathcal{V}_{\chi_k(p)} \subseteq \mathcal{V}_\alpha$, *for every* $p \in V_k$, *and* $R_k, X_k \vdash_{\mathrm{i}} \beta \leftrightarrow \chi_k(\beta)$, *for every* β.
(iv) At every step after (S2), $w \models R_k$, *for every* $w \in W$.
(v) At every step after (S2), r_k *is the root of* $\mathcal{K}(W)$ *and* $r_k \Vdash R_k$ *and* $r_k \nVdash g$.

$\lambda_0 \ = (\tilde{p}_0 \to \bot) \to \tilde{p}_1 \qquad \lambda_1 \ = (a \to \bot) \to \tilde{p}_0$
$\lambda_2 \ = (a \to \tilde{p}_3) \to \tilde{p}_4 \qquad \lambda_3 \ = (a \to \bot) \to \tilde{p}_3 \qquad \lambda_4 \ = (\tilde{p}_3 \to a) \to \tilde{p}_5$

$w_0 = \emptyset \quad w_1 = \{\tilde{g}, \tilde{p}_0, \tilde{p}_2\} \quad w_2 = \{a, \tilde{g}, \tilde{p}_1, \tilde{p}_2\} \quad w_3 = \{\tilde{p}_4\} \quad w_4 = \{\tilde{g}, \tilde{p}_0, \tilde{p}_2, \tilde{p}_4\}$
$w_5 = \{\tilde{g}, \tilde{p}_0, \tilde{p}_2, \tilde{p}_3, \tilde{p}_4\} \quad w_6 = \{a, \tilde{p}_5\} \quad w_7 = \{\tilde{p}_3, \tilde{p}_4\} \quad w_8 = \{\tilde{g}, \tilde{p}_0, \tilde{p}_2, \tilde{p}_3, \tilde{p}_4\}$
$w_9 = \{\tilde{p}_5\} \quad w_{10} = \{\tilde{p}_4\} \quad w_{11} = \{\tilde{g}, \tilde{p}_0, \tilde{p}_2, \tilde{p}_3, \tilde{p}_4\}$

$\chi_0 = [\tilde{g} \mapsto \neg a \vee \neg\neg a,\ \tilde{p}_0 \mapsto \neg a,\ \tilde{p}_1 \mapsto \neg\neg a,\ \tilde{p}_2 \mapsto \neg a \vee \neg\neg a]$
$\chi_1 = [\tilde{p}_3 \mapsto \neg a,\ \tilde{p}_4 \mapsto a \to \neg a,\ \tilde{p}_5 \mapsto \neg a \to a]$

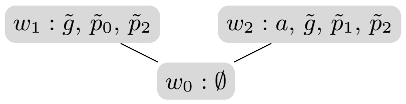
	@SAT	Answer	W	λ s.t. $w \not\rhd_W \lambda$
Start	(1) $R_0 \vdash_c \tilde{g}$?	No(w_0)	$\underline{w_0}$	$\underline{\lambda_0}, \lambda_1$
	(2) $R_0, w_0, \tilde{p}_0 \vdash_c \bot$?	No(w_1)	w_1	$\emptyset$
			$\underline{w_0}$	$\underline{\lambda_1}$
	(3) $R_0, w_0, a \vdash_c \bot$?	No(w_2)	w_2	$\emptyset$
			w_1	$\emptyset$
			w_0	$\emptyset$
Semantic failure		**Learned axiom**: $\psi_0 = (a \to \neg a) \vee (\neg a \to a)$		
SRest 1	(4) $R_1 \vdash_c \tilde{g}$?	No(w_3)	$\underline{w_3}$	$\underline{\lambda_0}, \lambda_1, \lambda_3, \lambda_4$
	(5) $R_1, w_3, \tilde{p}_0 \vdash_c \bot$?	No(w_4)	$\underline{w_4}$	$\lambda_3, \underline{\lambda_4}$
			w_3	$\lambda_1, \lambda_3, \lambda_4$
	(6) $R_1, w_4, \tilde{p}_3 \vdash_c a$?	No(w_5)	w_5	$\emptyset$
			$\underline{w_4}$	$\underline{\lambda_3}$
			w_3	λ_1, λ_3
	(7) $R_1, w_4, a \vdash_c \bot$?	Yes($\{a, \tilde{p}_0\}$)	$\psi_1 = \tilde{p}_0 \to \tilde{p}_3$	
BRest 2	(8) $R_2 \vdash_c \tilde{g}$?	No(w_6)	$\underline{w_6}$	$\underline{\lambda_0}$
	(9) $R_2, w_6, \tilde{p}_0 \vdash_c \bot$?	Yes($\{a, \tilde{p}_0\}$)	$\psi_2 = a \to \tilde{p}_1$	
BRest 3	(10) $R_3 \vdash_c \tilde{g}$?	No(w_7)	$\underline{w_7}$	$\underline{\lambda_0}, \lambda_1$
	(11) $R_3, w_7, \tilde{p}_0 \vdash_c \bot$?	No(w_8)	w_8	$\emptyset$
			$\underline{w_7}$	$\underline{\lambda_1}$
	(12) $R_3, w_7, a \vdash_c \bot$?	Yes($\{a, \tilde{p}_3\}$)	$\psi_3 = \tilde{p}_3 \to \tilde{p}_0$	
BRest 4	(13) $R_4 \vdash_c \tilde{g}$?	No(w_9)	$\underline{w_9}$	$\underline{\lambda_0}, \lambda_1, \lambda_2, \lambda_3$
	(14) $R_4, w_9, \tilde{p}_0 \vdash_c \bot$?	Yes($\{\tilde{p}_0, \tilde{p}_5\}$)	$\psi_4 = \tilde{p}_5 \to \tilde{p}_1$	
BRest 5	(15) $R_5 \vdash_c \tilde{g}$?	No(w_{10})	$\underline{w_{10}}$	$\underline{\lambda_0}, \lambda_1, \lambda_3, \lambda_4$
	(16) $R_5, w_{10}, \tilde{p}_0 \vdash_c \bot$?	No(w_{11})	w_{11}	$\emptyset$
			$\underline{w_{10}}$	$\underline{\lambda_1}, \lambda_3$
	(17) $R_5, w_{10}, a \vdash_c \bot$?	Yes($\{a, \tilde{p}_4\}$)	$\psi_5 = \tilde{p}_4 \to \tilde{p}_0$	
BRest 6	(18) $R_6 \vdash_c \tilde{g}$?	Yes($\emptyset$)	**Proved**	

Fig. 4. Computation of `intuitRIL(`$\neg a \vee \neg\neg a$, GL`)`.

$$\mathcal{T}_0 = \quad \dfrac{R_0, X_0 \Rightarrow g}{\Rightarrow \alpha}\ \mathrm{Claus}_0(g,\chi')$$

$$\mathcal{T}_k \ = \begin{array}{c} \dfrac{R_{k-1}, A \vdash_{\mathrm{c}} b \quad R_k, X_k \Rightarrow g}{R_{k-1}, X_{k-1} \Rightarrow g}\ \mathrm{cpl}_1(\lambda) \\ \vdots\ \mathcal{T}_{k-1} \\ \Rightarrow \alpha \end{array} \qquad \mathcal{T}_k \ = \begin{array}{c} \dfrac{R_k, X_k \Rightarrow g}{R_{k-1}, X_{k-1} \Rightarrow g}\ \mathrm{Claus}_1(\psi,\chi') \\ \vdots\ \mathcal{T}_{k-1} \\ \Rightarrow \alpha \end{array}$$

if $k > 0$ and iteration $k-1$ ends with a basic restart (thus $X_k = X_{k-1}$)

if $k > 0$ and iteration $k-1$ ends with a semantic restart

Fig. 5. Definition of $\mathcal{T}_k$ $(k \geq 0)$.

(vi) *At Step (S4), $r_k \Vdash R_k \cup X_k \cup \Psi_k$ and $r_k \nVdash g$ (in $\mathcal{K}(W)$).*

(vii) *Assume that iteration k ends with a basic restart and let φ be the learned basic clause. For every $\varphi' \in \Phi_k$, $\varphi \not\sim_{\mathrm{c}} \varphi'$.*

(viii) *Assume that iteration k ends with a semantic restart and let ψ be the learned axiom. For every $\psi' \in \Psi_k$, $\chi_k(\psi) \not\sim_{\mathrm{i}} \chi_k(\psi')$.*

Proof. We only sketch the proof of the non-trivial points.

(iii). By Lemma 2 applied to $\mathcal{T}_k$.

(v). Every interpretation M generated at Step (S6) is a superset of r_k, thus after Step (S2) r_k is the minimum element of W and the root of $\mathcal{K}(W)$. By (iv) and Prop. 2(i), $r_k \Vdash R_k$. Since $g \notin r_k$, we get $r_k \nVdash g$.

(vi). At Step (S4), $w \rhd_W \lambda$ for every $w \in W$ and $\lambda \in X_k$. Since $(X_k)^\star \subseteq R_k$, by Prop. 2(ii) we get $r_k \Vdash X_k$. Let $\psi \in \Psi_k$; then, ψ has been learned at some iteration $k' < k$. Let (R', X', χ') be the output of `Clausify(`ψ`,`V`)` at Step (S5) of iteration k' . Since $R' \subseteq R_k$ and $X' \subseteq X_k$, it holds that $r_k \Vdash R' \cup X'$. By (P1) $R', X' \vdash_{\mathrm{i}} \psi$, hence $r_k \Vdash \psi$, which proves $r_k \Vdash \Psi_k$.

(vii). Let $\varphi' \in \Phi_k$; we show that $\varphi \not\sim_{\mathrm{c}} \varphi'$. Let $\varphi = \bigwedge(A \setminus \{a\}) \to c$; then, there are $w \in W$ and $\lambda = (a \to b) \to c \in X_k$ such that $\langle w, \lambda \rangle$ has been selected at Step (S3) and the outcome of `satProve(`s`,`$w \cup \{a\}$`,`b`)` at Step (S6) is Yes(A). Note that $w \not\rhd_W \lambda$, hence $c \notin w$; since $A \subseteq w \cup \{a\}$, we get $w \not\models \varphi$. On the other hand, $w \models \varphi'$, since $\varphi' \in \Phi_k$ and $\Phi_k \subseteq R_k$. We conclude $\varphi \not\sim_{\mathrm{c}} \varphi'$.

(viii). Let $\psi' \in \Psi_k$ and let $\mathcal{K}(W)$ be the model obtained at Step (S4) of iteration k. By (iii) $R_k, X_k \vdash_{\mathrm{i}} \psi \leftrightarrow \chi_k(\psi)$ and $R_k, X_k \vdash_{\mathrm{i}} \psi' \leftrightarrow \chi_k(\psi')$. Since $r_k \nVdash \psi$ and $r_k \Vdash \psi'$ (indeed, $\psi' \in \Psi_k$ and $r_k \Vdash \Psi_k$) and $r_k \Vdash R_k \cup X_k$, we get $r_k \nVdash \chi_k(\psi)$ and $r_k \Vdash \chi_k(\psi')$. We conclude $\chi_k(\psi) \not\sim_{\mathrm{i}} \chi_k(\psi')$. □

The following proposition proves the partial correctness of `intuitRIL`:

Proposition 3. `intuitRIL(`α`,`L`)` *satisfies properties (Q1) and (Q2).*

Proof. Let us assume that the computation ends at iteration k with output Ψ_α. Then, the call to the SAT-solver at Step (S0) yields Yes($\emptyset$), meaning that

$R_k \vdash_{\mathrm{c}} g$. We can build the following C_L-derivation $\mathcal{D}$ of $\Rightarrow \alpha$:

$$\mathcal{D} \;=\; \begin{array}{c} \dfrac{R_k \vdash_{\mathrm{c}} g}{R_k, X_k \Rightarrow g}\,\mathrm{cpl}_0 \\ \vdots\; \mathcal{T}_k \\ \Rightarrow \alpha \end{array} \qquad\qquad \pi(\mathcal{D}) = \pi(\mathcal{T}_k) = \langle \varPsi_k, \chi_k \rangle$$

Note that $\varPsi_\alpha = \chi_k(\varPsi_k)$. Accordingly, by Prop. 1 we get (Q1).

Let us assume that the output is the model $\mathcal{K}(W)$, having root r. Then, $\mathcal{K}(W)$ is an L-model (otherwise, Step (S4) should have forced a semantic restart). By Lemma 3(vi) we get $r \Vdash R_0 \cup X_0$ and $r \nVdash g$. Since at Step (S0) we have clausified the formula $\alpha \leftrightarrow g$, by (P1) we get $R_0, X_0 \vdash_{\mathrm{i}} \alpha \leftrightarrow g$, which implies $r \Vdash \alpha \leftrightarrow g$. We conclude that $r \nVdash \alpha$, hence (Q2) holds. □

It seems challenging to provide a general proof of termination, and each logic must be treated apart. We can only state some general properties about the termination of the inner loop and of consecutive basic restarts.

Proposition 4. *(i) The inner loop is terminating.*
(ii) The number of consecutive basic restarts is finite.

Proof. Let us assume, by absurd, that the inner loop is not terminating. For every $j \geq 0$, by W_j we denote the value of W at Step (S3) of iteration j of the inner loop; note that the value of the variable V does not change during the iterations. We show that $W_j \subset W_{j+1}$, for every $j \geq 0$. At iteration j, the outcome of Step (S6) is $\mathrm{No}(M)$. Thus, there are $w \in W_j$ and $\lambda = (a \to b) \to c \in X$ such that the pair $\langle w, \lambda \rangle$ has been selected at Step (S3); accordingly, $w \not\triangleright_{W_j} \lambda$ and $w \cup \{a\} \subseteq M$ and $b \notin M$. We have $M \notin W_j$, otherwise we would get $w \triangleright_{W_j} \lambda$, a contradiction. Since $W_{j+1} = W_j \cup \{M\}$, this proves that $W_j \subset W_{j+1}$. We have shown that $W_0 \subset W_1 \subset W_2 \ldots$. This leads to a contradiction since, for every $j \geq 0$ and every $w \in W_j$, w is a subset of V and V is finite. We conclude that the inner loop is terminating, and this proves (i).

Let us assume, by contradiction, that there is an infinite sequence of consecutive basic restarts. Then, there is $n \geq 0$ such that, for every $k \geq n$, the iteration k of the main loop ends with a basic restart. Let φ_k be the clause learned at iteration k. Note that an iteration ending with a basic restart does not introduce new atoms, thus $\mathcal{V}_{\varphi_k} \subseteq V_n$ for every $k \geq n$ (where V_n is defined as in (†)). We get a contradiction, since V_n is finite and, by Lemma 3(vii), the clauses φ_k are pairwise non $\sim_{\mathrm{c}}$-equivalent; this proves (ii). □

Lemma 3(viii) guarantees that the learned axioms are pairwise distinct, but this is not sufficient to prove termination since in general we cannot set a bound on the size and on the number of learned axioms. In next section we present some relevant logics where the procedure is terminating.

5 Termination

Let GL = IPL + **lin** be the Gödel-Dummett logic presented in Ex. 1; we show that every call `intuitRIL(`α`,GL)` is terminating. To this aim, we exploit the bounding function $\mathrm{Ax}_{\mathrm{GL}}(\alpha)$ presented in the mentioned example.

Lemma 4. *Let us consider the computation of* `intuitRIL(`α`,GL)` *and assume that at iteration k of the main loop Step (S4) is executed and that the obtained model $\mathcal{K}(W)$ is not linear. Then, there exists $\psi \in \mathrm{Ax}_{\mathrm{GL}}(\alpha)$ such that $r_k \nVdash \psi$.*

Proof. Let us assume that $\mathcal{K}(W)$ has two distinct maximal worlds w_1 and w_2; note that $w_1 \subseteq V_k$ and $w_2 \subseteq V_k$ (with V_k defined as in (†)). We show that:
(a) $w_1 \cap \mathcal{V}_\alpha \neq w_2 \cap \mathcal{V}_\alpha$.

Suppose by contradiction $w_1 \cap \mathcal{V}_\alpha = w_2 \cap \mathcal{V}_\alpha$; let $p \in V_k$ and $\beta = \chi_k(p)$ (with χ_k defined as in (†)). By Lemma 3(iii), $R_k, X_k \vdash_{\mathrm{i}} p \leftrightarrow \beta$; by Lemma 3(vi) we get $w_1 \Vdash p \leftrightarrow \beta$ and $w_2 \Vdash p \leftrightarrow \beta$. Since $\mathcal{V}_\beta \subseteq \mathcal{V}_\alpha$ (see Lemma 3(iii)) and we are assuming $w_1 \cap \mathcal{V}_\alpha = w_2 \cap \mathcal{V}_\alpha$, it holds that $w_1 \Vdash \beta$ iff $w_2 \Vdash \beta$, thus $w_1 \Vdash p$ iff $w_2 \Vdash p$, namely $p \in w_1$ iff $p \in w_2$. Since p is any element of V_k, we get $w_1 = w_2$, a contradiction; this proves (a). By (a) there is $a \in \mathcal{V}_\alpha$ such that either $a \in w_1 \setminus w_2$ or $a \in w_2 \setminus w_1$. We consider the former case (the latter one is symmetric), corresponding to Case 1 in Fig. 6. We have $w_1 \Vdash a$ and $w_2 \Vdash \neg a$; setting $\psi = (a \to \neg a) \vee (\neg a \to a)$, we conclude $r_k \nVdash \psi$.

Assume that $\mathcal{K}(W)$ has only one maximal world; since it is not linear, there are three distinct worlds w_1, w_2, w_3 as in Case 2 in Fig. 6, namely: w_1 is an immediate successor of w_2 and w_3 (i.e., for $j \in \{2,3\}$, $w_j < w_1$ and, if $w_j < w$, then $w_1 \leq w$), $w_2 \nleq w_3$, $w_3 \nleq w_2$. Reasoning as in (a), we get:

(b) $w_2 \cap \mathcal{V}_\alpha \neq w_3 \cap \mathcal{V}_\alpha$. (c) $w_2 \cap \mathcal{V}_\alpha \subset w_1 \cap \mathcal{V}_\alpha$ and $w_3 \cap \mathcal{V}_\alpha \subset w_1 \cap \mathcal{V}_\alpha$.

By (b) there is $a \in \mathcal{V}_\alpha$ such that either $a \in w_2 \setminus w_3$ or $a \in w_3 \setminus w_2$. Let us consider the former case (the latter one is symmetric). By (c), there is $b \in \mathcal{V}_\alpha$ such that $b \in w_1 \setminus w_2$. If $b \in w_3$ (Case 2.1 in Fig. 6), we get $a \in w_2$, $b \notin w_2$, $a \notin w_3$, $b \in w_3$. Setting $\psi = (a \to b) \vee (b \to a)$, we conclude $r_k \nVdash \psi$. Finally, let us assume $b \notin w_3$ (Case 2.2). We have $\{a, b\} \subseteq w_1$, $a \in w_2$, $b \notin w_2$, $a \notin w_3$ and $b \notin w_3$. It is easy to check that $w_3 \Vdash a \to b$ (recall that $w_3 < w$ implies $w_1 \leq w$), thus $w_3 \nVdash (a \to b) \to a$. On the other hand $w_2 \nVdash a \to (a \to b)$. Setting $\psi = (a \to (a \to b)) \vee ((a \to b) \to a)$, we get $r_k \nVdash \psi$. □

We exploit Lemma 4 to implement Step (S4). If $\mathcal{K}(W)$ is linear, then $\mathcal{K}(W)$ is a GL-model and we are done. Otherwise, the proof of Lemma 4 hints an effective method to select an instance ψ of **lin** from $\mathrm{Ax}_{\mathrm{GL}}(\alpha)$.

Proposition 5. *The computation of* `intuitRIL(`α`,GL)` *is terminating.*

Proof. Assume that `intuitRIL(`α`,GL)` is not terminating. Since the number of iterations of the inner loop and of the consecutive basic restarts is finite (see Prop. 4), Step (S4) must be executed infinitely many times. This leads to a contradiction, since the axioms selected at Step (S4) are pairwise distinct (see Lemma 3(viii)) and such axioms are chosen from the finite set $\mathrm{Ax}_{\mathrm{GL}}(\alpha)$. □

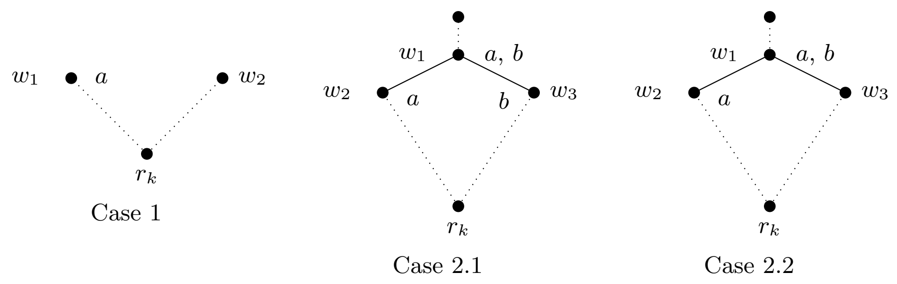


Fig. 6. Proof of Lemma 4, case analysis.

As a corollary, we get that $\mathrm{Ax}_{\mathrm{GL}}(\alpha)$ is a bounding function for GL:

Proposition 6. *If α is* GL*-valid, there is $\Psi_\alpha \subseteq \mathrm{Ax}_{\mathrm{GL}}(\alpha)$ such that $\Psi_\alpha \vdash_{\mathrm{i}} \alpha$.*

Other proof-search strategies for GL are discussed in [10,14]. This technique can be extended to other notable intermediate logics. Among these, we recall the logics GL_n (Gödel Logic of depth n), obtained by adding to GL the axioms $\mathbf{bd}_n$ (bounded depth) where: $\mathbf{bd}_0 = a_0 \vee \neg a_0$, $\mathbf{bd}_{n+1} = a_{n+1} \vee (a_{n+1} \to \mathbf{bd}_n)$. Semantically, GL_n is the logic characterized by linear frames having depth at most n. We are not able to prove termination for the logics $\mathrm{IPL} + \mathbf{bd}_n$, but we can implement the following terminating strategy for GL_n. Let $\mathcal{K}(W)$ be the model obtained at Step (S4) of the computation of `intuitRIL(`α,GL_n`)`:

- If $\mathcal{K}(W)$ is not linear, we select the axiom ψ from $\mathrm{Ax}_{\mathrm{GL}}(\alpha)$.
- Otherwise, assume that $\mathcal{K}(W)$ is linear but not a GL_n-model. Then, $\mathcal{K}(W)$ contains a chain of worlds $w_0 \subset w_1 \subset \cdots \subset w_{n+1}$. The crucial point is that $w_{j+1} \setminus w_j$ contains at least a propositional variable from $\mathcal{V}_\alpha$, for every $0 \leq j \leq n$. Thus, we can choose a proper renaming of $\mathbf{bd}_n$ as ψ.

Another terminating logic is the Jankov Logic (see Ex. 3); actually, also in this case the learned axiom can be chosen by renaming the **wem** axiom. In general, all the logics BTW_n (Bounded Top Width, at most n maximal worlds, see [2]) are terminating. An intriguing case is Scott Logic ST [2]: even though the class of ST-frames is not first-order definable, we can implement a learning procedure for ST-axioms arguing as in [7] (see Sec. 2.5.2). Some of the mentioned logics have been implemented in `intuitRIL`[3].

One may wonder whether this method can be applied to other non-classical logics or to fragments of predicate logics (these issues have been already raised in the seminal paper [4]). A significant work in this direction is [11], where the procedure has been applied to some modal logics. However, the main difference with the original approach is that it is not possible to use a single SAT-solver, but one needs a supply of SAT-solvers. This is primarily due to the fact that forcing relation of modal Kripke models is not persistent; thus worlds are loosely related and must be handled by independent solvers.

[3] Available at `https://github.com/cfiorentini/intuitRIL`.

References

1. Avellone, A., Miglioli, P., Moscato, U., Ornaghi, M.: Generalized tableau systems for intemediate propositional logics. In: Galmiche, D. (ed.) TABLEAUX '97. LNCS, vol. 1227, pp. 43–61. Springer (1997)
2. Chagrov, A.V., Zakharyaschev, M.: Modal Logic, Oxford logic guides, vol. 35. Oxford University Press (1997)
3. Ciabattoni, A., Lang, T., Ramanayake, R.: Bounded-analytic sequent calculi and embeddings for hypersequent logics. J. Symb. Log. **86**(2), 635–668 (2021)
4. Claessen, K., Rosén, D.: SAT Modulo Intuitionistic Implications. In: Davis, M., Fehnker, A., McIver, A., Voronkov, A. (eds.) LPAR-20. LNCS, vol. 9450, pp. 622–637. Springer (2015)
5. Dyckhoff, R.: Contraction-free sequent calculi for intuitionistic logic. J. Symb. Log. **57**(3), 795–807 (1992)
6. Ferrari, M., Fiorentini, C., Fiorino, G.: fCube: An Efficient Prover for Intuitionistic Propositional Logic. In: Fermüller, C.G., Voronkov, A. (eds.) LPAR-17. LNCS, vol. 6397, pp. 294–301. Springer (2010)
7. Fiorentini, C.: Kripke Completeness for Intermediate Logics. Ph.D. thesis, Università degli Studi di Milano (2000)
8. Fiorentini, C.: Efficient sat-based proof search in intuitionistic propositional logic. In: Platzer, A., Sutcliffe, G. (eds.) CADE 28. LNCS, vol. 12699, pp. 217–233. Springer (2021)
9. Fiorentini, C., Goré, R., Graham-Lengrand, S.: A Proof-Theoretic Perspective on SMT-Solving for Intuitionistic Propositional Logic. In: Cerrito, S., Popescu, A. (eds.) TABLEAUX 2019. LNCS, vol. 11714, pp. 111–129. Springer (2019)
10. Fiorino, G.: Terminating calculi for propositional Dummett logic with subformula property. J. Autom. Reason. **52**(1), 67–97 (2014)
11. Goré, R., Kikkert, C.: CEGAR-tableaux: Improved modal satisfiability via modal clause-learning and SAT. In: Das, A., Negri, S. (eds.) TABLEAUX 2021. LNCS, vol. 12842, pp. 74–91. Springer (2021)
12. Goré, R., Thomson, J., Wu, J.: A History-Based Theorem Prover for Intuitionistic Propositional Logic Using Global Caching: IntHistGC System Description. In: Demri, S., Kapur, D., Weidenbach, C. (eds.) IJCAR 2014. LNCS, vol. 8562, pp. 262–268. Springer (2014)
13. Jankov, V.: The calculus of the weak "law of excluded middle". Mathematics of the USSR **8**, 648–650 (1968)
14. Larchey-Wendling, D.: Gödel-dummett counter-models through matrix computation. Electron. Notes Theor. Comput. Sci. **125**(3), 137–148 (2005)
15. Lifschitz, V., Pearce, D., Valverde, A.: Strongly equivalent logic programs. ACM Trans. Comput. Log. **2**(4), 526–541 (2001)
16. Nieuwenhuis, R., Oliveras, A., Tinelli, C.: Solving SAT and SAT Modulo Theories: From an abstract Davis–Putnam–Logemann–Loveland procedure to DPLL(T). J. ACM **53**(6), 937–977 (2006)
17. Schmidt, R.A., Tishkovsky, D.: Automated synthesis of tableau calculi. Log. Methods Comput. Sci. **7**(2) (2011)
18. Troelstra, A.S., Schwichtenberg, H.: Basic proof theory, Second Edition, Cambridge tracts in theoretical computer science, vol. 43. Cambridge University Press (2000)

A Appendix

A.1 Clausify

We present the `Clausify` procedure we exploit in this paper; it is similar to the one described in [4], with slight modifications in order to match (C3). The computation of `Clausify(`α`,`V`)` consists of a sequence of rewriting steps in order to turn α into an equivalent set of clauses R and implication clauses X satisfying properties (C1)–(C3). Some steps require the introduction of new propositional variables to represent compound formulas; to implement this, we exploit a global cache storing the correspondence between formulas and new variables. New variables have the form $\tilde{p}_k$ (with $k \geq 0$) and it is assumed that each $\tilde{p}_k$ does not belong to the set V (second parameter of `Clausify`); we point out that at any step the formulas in the cache can contain variables from V and new variables introduced in previous steps. We write $\alpha \mapsto \tilde{p}_k$ to mean that, in the current cache, the formula α is represented by the new variable $\tilde{p}_k$. We point out that the first query $\alpha \mapsto \tilde{p}_k$ introduces the new variable $\tilde{p}_k$ and the cache is updated accordingly; next queries simply return $\tilde{p}_k$ (no update is needed). The rewriting steps are performed by the auxiliary recursive function `ClausAux` displayed in Fig. 7; the call `ClausAux(`α`)` returns the pair $\langle R, X\rangle$ corresponding to the clausification of α. To avoid an overloading notation, the management of the global cache is left understood. We also perform further optimizations, not detailed here, for instance: before clausifying the formulas α, we apply boolean simplifications (e.g., $\alpha_1 \to \alpha_2 \vee \bot$ is simplified as $\alpha_1 \to \alpha_2$), we avoid to clausify the same formula $\tilde{p}_k \leftrightarrow \alpha$ more than once. The main difference w.r.t. the original procedure [4] (implemented in the provers `intuit` and `intuitR`) is that, when the correspondence $\alpha \mapsto \tilde{p}_k$ is used, we require the clausification of the formula $\tilde{p}_k \leftrightarrow \alpha$ setting a strong equivalence between α and the associated variable $\tilde{p}_k$; the original approach introduces a looser correspondence. For instance, let us consider the case $\alpha = (\alpha_1 \to \alpha_2) \to \alpha_3$ and let us assume $\alpha_1 \mapsto \tilde{p}_1$, $\alpha_2 \mapsto \tilde{p}_2$ and $\alpha_3 \mapsto \tilde{p}_3$. We require the clausification of the formulas $\tilde{p}_1 \leftrightarrow \alpha_1$, $\tilde{p}_2 \leftrightarrow \alpha_2$ and $\tilde{p}_3 \leftrightarrow \alpha_3$; instead, in [4] the clausified formulas are $\tilde{p}_1 \to \alpha_1$, $\alpha_2 \to \tilde{p}_2$ and $\tilde{p}_3 \to \alpha_3$. When all the reductions have been performed by `ClausAux`, the substitution χ can be computed by inspecting the global cache.

A.2 Omitted proofs

Lemma 1. *(i)* $R \vdash_{\mathrm{i}} g$ *iff* $R \vdash_{\mathrm{c}} g$*, for every set of clauses* R *and every atom* g.
(ii) $X \vdash_{\mathrm{i}} b \to c$*, for every* $b \to c \in (X)^\star$.
(iii) $\Gamma \vdash_{\mathrm{i}} \alpha$ *iff* $\alpha \leftrightarrow g, \Gamma \vdash_{\mathrm{i}} g$*, where* $g \notin \mathcal{V}_{\Gamma,\alpha}$.

Proof. *(i)* (the proof was originally given in [9]). The proof that $R \vdash_{\mathrm{i}} g$ implies $R \vdash_{\mathrm{c}} g$ is trivial. Conversely, suppose that $R \nvdash_{\mathrm{i}} g$. Then, there is a Kripke model $\langle W, \leq, r, \vartheta\rangle$ such that $r \Vdash \bigwedge R$ and $r \nVdash g$. Let $\varphi \in R$ and assume that $\varphi = \bigwedge A_1 \to \bigvee A_2$ (if $\varphi = \bigvee A_2$, the proof is similar). The valuation $\vartheta(r)$ either has $A_2 \cap \vartheta(r) \neq \emptyset$ or $A_2 \cap \vartheta(r) = \emptyset$. If $A_2 \cap \vartheta(r) \neq \emptyset$ then φ is classically true at r. If $A_2 \cap \vartheta(r) = \emptyset$, then reflexivity demands $A_1 \not\subseteq \vartheta(r)$, as otherwise $r \not\Vdash \varphi$, contradicting our assumption. Again, φ is classically true at r. That is, r by itself is a classical model that also makes all the clauses in R true and g false, so $R \nvdash_{\mathrm{c}} g$. This proves that $R \vdash_{\mathrm{c}} g$ implies $R \vdash_{\mathrm{i}} g$. Notice that this proof only works because R contains clauses.

(ii) is trivial.

(iii). Let $\Gamma \vdash_{\mathrm{i}} \alpha$; then $\alpha \leftrightarrow g, \Gamma \vdash_{\mathrm{i}} \alpha$, hence $\alpha \leftrightarrow g, \Gamma \vdash_{\mathrm{i}} g$. Conversely, assume $\Gamma \nvdash_{\mathrm{i}} \alpha$. There is a model $\mathcal{K}$ such that $r \Vdash \Gamma$ and $r \nVdash \alpha$, with r the root of $\mathcal{K}$. Let $\mathcal{K}'$ be obtained from $\mathcal{K}$ by changing the valuation on g so that g behaves like α (i.e., $w \Vdash g \leftrightarrow \alpha$ for every world w). Since $g \notin \mathcal{V}_{\Gamma,\alpha}$, in $\mathcal{K}'$ it holds that $r \Vdash \Gamma$ and $r \nVdash \alpha$, hence $r \nVdash g$. We conclude $\alpha \leftrightarrow g, \Gamma \nvdash_{\mathrm{i}} g$. □

Proposition 2. *Let $\mathcal{K}(W)$ be the model generated by W and let $w \in W$. Let φ be a clause and $\lambda = (a \to b) \to c$ an implication clause.*

(i) If $w' \models \varphi$, for every $w' \in W$ such that $w \le w'$, then $w \Vdash \varphi$.
(ii) If $w' \models b \to c$ and $w' \rhd_W \lambda$, for every $w' \in W$ such that $w \le w'$, then $w \Vdash \lambda$.

Proof. Let $\varphi = \bigwedge A_1 \to \bigvee A_2$ and assume that $w \nVdash \varphi$. There exists $w' \in W$ such that $w \le w'$ and $w' \Vdash \bigwedge A_1$ and $w' \nVdash \bigvee A_2$. By definition of $\mathcal{K}(W)$, it follows that $A_1 \subseteq w'$ and $A_2 \cap w' = \emptyset$, hence $w' \not\models \varphi$, and this proves (i). The proof of (i) in the case $\varphi = \bigvee A_2$ is similar.

Let us assume that $w' \models b \to c$ and $w' \rhd_W \lambda$, for every $w' \in W$ such that $w \le w'$; suppose, by contradiction, $w \nVdash \lambda$. There exists $w' \in W$ such that $w \le w'$ and $w' \Vdash a \to b$ and $w' \nVdash c$, namely $w' \not\models c$. Since $w' \models b \to c$, we get $w' \not\models b$, which implies $w' \nVdash b$. By the fact that $w' \Vdash a \to b$ and $w' \nVdash b$, we get $w' \nVdash a$; thus, $a \notin w'$ and $b \notin w'$ and $c \notin w'$. Since $w' \rhd_W \lambda$, there exists $w'' \in W$ such that $w' \le w''$ and $a \in w''$ and $b \notin w''$. It follows that $w'' \Vdash a$ and $w'' \nVdash b$, in contradiction with the fact that $w' \Vdash a \to b$. We conclude $w \Vdash \lambda$. □

Lemma 2. *Let $\mathcal{T} = \mathcal{T}(\alpha; R, X \Rightarrow g)$ and let $\pi(\mathcal{T}) = \langle \Psi, \chi \rangle$.*

(i) $\mathcal{V}_{\chi(p)} \subseteq \mathcal{V}_\alpha$, for every $p \in \mathcal{V}_\mathcal{T}$.
(ii) $R, X \vdash_{\mathrm{i}} \beta \leftrightarrow \chi(\beta)$, for every formula β.
(iii) If $R, X, \Gamma \vdash_{\mathrm{i}} g$ and $\mathcal{V}_\Gamma \subseteq \mathcal{V}_\alpha$, then $\Gamma, \chi(\Psi) \vdash_{\mathrm{i}} \alpha$.

Proof. We proceed by induction on the depth of $\mathcal{T}$. The base case is:

$$\mathcal{T} = \quad \cfrac{\overbrace{R', (X)^\star}^{R},\ X \Rightarrow g}{\Rightarrow \alpha}\ \mathrm{Claus}_0(g, \chi') \qquad \begin{array}{l} g \notin \mathcal{V}_\alpha \\ (R', X, \chi') = \texttt{Clausify}(\alpha \leftrightarrow g,\ \mathcal{V}_{\alpha,g}) \end{array}$$

We point out that $\Psi = \emptyset$ and $\chi = [g \mapsto \alpha] \cdot \chi'$. By conditions (C2), (C3) and property (P1) we get:

(a) $\mathcal{V}_{\chi'(p)} \subseteq \mathcal{V}_{\alpha,g}$, for every $p \in \mathcal{V}_{\alpha,g} \cup \mathcal{V}_{R',X}$.
(b) $R', X \vdash_{\mathrm{i}} \beta \leftrightarrow \chi'(\beta)$, for every formula β, and $R', X \vdash_{\mathrm{i}} \alpha \leftrightarrow g$.

Note that $\mathcal{V}_{\mathcal{T}} = \mathcal{V}_{\alpha,g} \cup \mathcal{V}_{R',X}$; thus (i) follows from (a) and the definition of χ. Point (ii) is entailed by (b). Assume $R, X, \Gamma \vdash_{\mathrm{i}} g$ and $\mathcal{V}_\Gamma \subseteq \mathcal{V}_\alpha$. By Lemma 1(ii), it follows that $R', X, \Gamma \vdash_{\mathrm{i}} g$ and by (C1) we get $\alpha \leftrightarrow g, \Gamma \vdash_{\mathrm{i}} g$. By Lemma 1(iii) we conclude $\Gamma \vdash_{\mathrm{i}} \alpha$, thus (iii) holds. Now let

$$\mathcal{T} \;=\; \begin{array}{c} \dfrac{R', A \vdash_{\mathrm{c}} b \qquad \overbrace{R', \varphi}^{R},\, X \Rightarrow g}{R', X \Rightarrow g}\,\mathrm{cpl}_1(\lambda) \\ \vdots\; \mathcal{T}' \\ \Rightarrow \alpha \end{array} \qquad \begin{array}{l} \lambda = (a \to b) \to c \in X \\ A \subseteq \mathcal{V}_{R',X,g} \\ \varphi \;=\; \bigwedge(A \setminus \{a\}) \to c \end{array}$$

We point out that $\mathcal{T}' = \mathcal{T}(\alpha; R', X \Rightarrow g)$ and $\mathcal{V}_{\mathcal{T}} = \mathcal{V}_{\mathcal{T}'}$ and $\pi(\mathcal{T}) = \pi(\mathcal{T}') = \langle \Psi, \chi \rangle$. Points (i) and (ii) immediately follow by the induction hypothesis. Assume $R, X, \Gamma \vdash_{\mathrm{i}} g$ and $\mathcal{V}_\Gamma \subseteq \mathcal{V}_\alpha$. Since $R', A \vdash_{\mathrm{c}} b$, by Lemma 1(i) we get $R', A \vdash_{\mathrm{i}} b$, which implies $R', \lambda \vdash_{\mathrm{i}} \varphi$, hence $R', X \vdash_{\mathrm{i}} \varphi$. Since $R', \varphi, X, \Gamma \vdash_{\mathrm{i}} g$, we get $R', X, \Gamma \vdash_{\mathrm{i}} g$. By the the induction hypothesis we conclude $\Gamma, \chi(\Psi) \vdash_{\mathrm{i}} \alpha$, hence (iii) holds. Let

$$\mathcal{T} \;=\; \begin{array}{c} \dfrac{\overbrace{R', R'', (X'')^\star}^{R},\, \overbrace{X', X''}^{X} \Rightarrow g}{R', X' \Rightarrow g}\,\mathrm{Claus}_1(\psi, \chi'') \\ \vdots\; \mathcal{T}' \\ \Rightarrow \alpha \end{array} \qquad \begin{array}{l} V' \;=\; \mathcal{V}_{R',X',g} \\ \psi \in \mathrm{Ax}(L, V') \\ (R'', X'', \chi'') = \texttt{Clausify}(\psi, V') \end{array}$$

Note that $\mathcal{T}' = \mathcal{T}(\alpha; R', X' \Rightarrow g)$ and $\mathcal{V}_{\mathcal{T}} = \mathcal{V}_{\mathcal{T}'} \cup \mathcal{V}_{R'',X''}$. Let $\pi(\mathcal{T}') = \langle \Psi', \chi' \rangle$; then $\Psi = \Psi' \cup \{\psi\}$ and $\chi = \chi' \cdot \chi''$. By conditions (C2) and (C3) of `Clausify` and the induction hypothesis, we get:

(c) $\mathcal{V}_{\chi''(p)} \subseteq V'$, for every $p \in V' \cup \mathcal{V}_{R'',X''}$.
(d) $\mathcal{V}_{\chi'(p)} \subseteq \mathcal{V}_\alpha$, for every $p \in \mathcal{V}_{\mathcal{T}'}$.
(e) $R', X' \vdash_{\mathrm{i}} \beta \leftrightarrow \chi'(\beta)$ and $R'', X'' \vdash_{\mathrm{i}} \beta \leftrightarrow \chi''(\beta)$, for every formula β.

Since $\chi = \chi' \cdot \chi''$, by (c)–(e) we get (i) and (ii). Assume $R, X, \Gamma \vdash_{\mathrm{i}} g$ and $\mathcal{V}_\Gamma \subseteq \mathcal{V}_\alpha$. Reasoning as in the base case, we get $R', X', \Gamma, \psi \vdash_{\mathrm{i}} g$. By the induction hypothesis (point (ii)), it follows that $R', X' \vdash_{\mathrm{i}} \psi \leftrightarrow \chi'(\psi)$, hence $R', X', \Gamma, \chi'(\psi) \vdash_{\mathrm{i}} g$. We remark that $\mathcal{V}_{\chi'(\psi)} \subseteq \mathcal{V}_\alpha$ (see (d)). Let $\Gamma' = \Gamma \cup \{\chi'(\psi)\}$; since $R', X', \Gamma' \vdash_{\mathrm{i}} g$ and $\mathcal{V}_{\Gamma'} \subseteq \mathcal{V}_\alpha$, we can apply the induction hypothesis (point (iii)) and claim that $\Gamma', \chi'(\Psi') \vdash_{\mathrm{i}} g$, namely $\Gamma, \chi'(\Psi) \vdash_{\mathrm{i}} g$. One can easily check that $\mathcal{V}_\Psi \subseteq V'$; since $\chi = \chi' \cdot \chi''$ and $\mathrm{Dom}(\chi'') = \mathcal{V}_{R'',X''} \setminus V'$ (see (C2)), we get $\chi(\Psi) = \chi'(\Psi)$. Thus $\Gamma, \chi(\Psi) \vdash_{\mathrm{i}} g$, which proves (ii). $\square$

α	$\langle R, X\rangle$
$a \in \mathcal{V} \cup \{\bot\}$	$R = \{a\} \quad X = \emptyset$
$a_1 \wedge \cdots \wedge a_n \to b_1 \vee \cdots \vee b_m$ $a_i, b_j \in \mathcal{V} \cup \{\bot\}$	$R = \{\, a_1 \wedge \cdots \wedge a_n \to b_1 \vee \cdots \vee b_m \,\} \qquad X = \emptyset$
$(a \to b) \to c$ $\{a, b, c\} \subseteq \mathcal{V} \cup \{\bot\}$	$R = \emptyset \qquad X = \{\, (a \to b) \to c \,\}$
$\alpha_1 \wedge \alpha_2$	$\texttt{let}\ \langle R_k, X_k\rangle = \texttt{ClausAux(}\alpha_k\texttt{)},\ k \in \{1, 2\}$ $R = R_1 \cup R_2 \qquad X = X_1 \cup X_2$
$a_1 \vee \cdots \vee a_n \vee \alpha_1 \vee \cdots \vee \alpha_m$ $a_i \in \mathcal{V} \cup \{\bot\}$	$\texttt{let}\ \alpha_1 \mapsto \tilde{p}_1, \ldots, \alpha_m \mapsto \tilde{p}_m$ $\texttt{let}\ \langle R_k, X_k\rangle = \texttt{ClausAux}\ (\tilde{p}_k \leftrightarrow \alpha_k),\ k \in \{1, \ldots, m\}$ $R = R_1 \cup \cdots \cup R_m \cup \{a_1 \vee \cdots \vee a_n \vee \tilde{p}_1 \vee \cdots \vee \tilde{p}_m\}$ $X = X_1 \cup \cdots \cup X_m$
$(\alpha_1 \to \alpha_2) \to \alpha_3$	$\texttt{let}\ \alpha_1 \mapsto \tilde{p}_1,\ \alpha_2 \mapsto \tilde{p}_2,\ \alpha_3 \mapsto \tilde{p}_3$ $\texttt{let}\ \langle R_k, X_k\rangle = \texttt{ClausAux(}\tilde{p}_k \leftrightarrow \alpha_k\texttt{)},\ k \in \{1, 2, 3\}$ $R = R_1 \cup R_2 \cup R_3 \quad X = X_1 \cup X_2 \cup X_3 \cup \{\, (\tilde{p}_1 \to \tilde{p}_2) \to \tilde{p}_3 \,\}$
$\alpha_1 \vee \alpha_2 \to \beta$	$\texttt{let}\ \langle R_k, X_k\rangle = \texttt{ClausAux(}\alpha_k \to \beta\texttt{)},\ k \in \{1, 2\}$ $R = R_1 \cup R_2 \qquad X = X_1 \cup X_2$
$\alpha \to \beta_1 \wedge \beta_2$	$\texttt{let}\ \langle R_k, X_k\rangle = \texttt{ClausAux(}\alpha \to \beta_k\texttt{)},\ k \in \{1, 2\}$ $R = R_1 \cup R_2 \qquad X = X_1 \cup X_2$
$\alpha \to (\beta \to \gamma)$	$\langle R, X\rangle = \texttt{ClausAux(}\alpha \wedge \beta \to \gamma\texttt{)}$
$\alpha \to (\beta_1 \leftrightarrow \beta_2)$	$\texttt{let}\ \langle R_1, X_1\rangle = \texttt{ClausAux(}\alpha \wedge \beta_1 \to \beta_2\texttt{)}$ $\texttt{let}\ \langle R_2, X_2\rangle = \texttt{ClausAux(}\alpha \wedge \beta_2 \to \beta_1\texttt{)}$ $R = R_1 \cup R_2 \qquad X = X_1 \cup X_2$
$a_1 \wedge \cdots \wedge a_n \wedge \alpha_1 \wedge \cdots \wedge \alpha_m \to \beta$ $a_i \in \mathcal{V} \cup \{\bot\}$	$\texttt{let}\ \alpha_1 \mapsto \tilde{p}_1, \ldots, \alpha_m \mapsto \tilde{p}_m$ $\texttt{let}\ \gamma = a_1 \wedge \cdots \wedge a_n \wedge \tilde{p}_1 \wedge \cdots \wedge \tilde{p}_m \to \beta$ $\texttt{let}\ \langle R_0, X_0\rangle = \texttt{ClausAux(}\gamma\texttt{)}$ $\texttt{let}\ \langle R_k, X_k\rangle = \texttt{ClausAux(}\tilde{p}_k \leftrightarrow \alpha_k\texttt{)},\ k \in \{1, \ldots, m\}$ $R = R_0 \cup \cdots \cup R_m \qquad X = X_0 \cup \cdots \cup X_m$
$\alpha \to b_1 \vee \cdots \vee b_n \vee \beta_1 \vee \cdots \vee \beta_m$ $b_i \in \mathcal{V} \cup \{\bot\}$	$\texttt{let}\ \beta_1 \mapsto \tilde{p}_1, \ldots, \beta_m \mapsto \tilde{p}_m$ $\texttt{let}\ \gamma = \alpha \to b_1 \vee \cdots \vee b_n \vee \tilde{p}_1 \vee \cdots \vee \tilde{p}_m$ $\texttt{let}\ \langle R_0, X_0\rangle = \texttt{ClausAux(}\gamma\texttt{)}$ $\texttt{let}\ \langle R_k, X_k\rangle = \texttt{ClausAux(}\tilde{p}_k \leftrightarrow \beta_k\texttt{)}\ k \in \{1, \ldots, m\}$ $R = R_0 \cup \cdots \cup R_m \qquad X = X_0 \cup \cdots \cup X_m$
$(\alpha \leftrightarrow \beta) \to \gamma$	$\langle R, X\rangle = \texttt{ClausAux(}(\alpha \to \beta) \wedge (\beta \to \alpha) \to \gamma\texttt{)}$
$\alpha \leftrightarrow \beta$	$\langle R, X\rangle = \texttt{ClausAux(}(\alpha \to \beta) \wedge (\beta \to \alpha)\texttt{)}$

Fig. 7. Definition of the auxiliary recursive procedure $\texttt{ClausAux(}\alpha\texttt{)}$