A Tableaux Calculus for Default Intuitionistic Logic

Valentin Cassano[1], Raul Fervari[1], Guillaume Hoffmann[1], Carlos Areces[1](✉), and Pablo F. Castro[2]

[1] CONICET and Universidad Nacional de Córdoba, Córdoba, Argentina
carlos.areces@gmail.com

[2] CONICET and Universidad Nacional de Río Cuarto, Río Cuarto, Argentina

Abstract. We build a Default Logic variant on Intuitionistic Propositional Logic and develop a sound, complete, and terminating, tableaux calculus for it. We also present an implementation of the calculus. We motivate and illustrate the technical elements of our work with examples.

1 Introduction

Non-monotonic formalisms have traditionally been defined with a classical semantics [20]. More recently –following the seminal work of Gabbay [19]– there is an interest in the interplay between non-monotonic formalisms and Intuitionistic Logic (IL). E.g., in [38] a formalization of a notion of non-monotonic implication capturing non-monotonic consequence based on IL is proposed; in [39] the work of Gabbay in [19] is revised; in [30] a characterization of answer sets for logic programs with nested expressions is offered in terms of provability in IL (generalizing earlier proposals of Pearce in [32,33]). The articles just mentioned have in common a study of the interplay between non-monotonic formalisms and IL from a theoretical perspective. Another interesting take on this interplay can be found in the area of Normative Systems or Legal Artificial Intelligence. E.g., in [22] a Description Logic built on IL is presented as a way to deal with conflicts present in laws and normative systems. These conflicts usually lead to logical inconsistencies when they are formally analyzed, bringing to the fore the need for an adequate semantics for negation in such a context. Another example is [31], where a construction of an I/O Logic –a general framework to study and reason about conditional norms [28]– is carried out on IL.

The interplay between non-monotonic formalisms and IL in normative systems is succinctly illustrated by the following motivating example (adapted from [26]). Let the possible outcomes of a trial be the verdicts of *guilty* or *not guilty*. A verdict of *guilty* is obtained when the evidence presented by the prosecution meets the so-called "beyond reasonable doubt" standard of proof. A verdict of *not guilty* is obtained when the evidence fails to meet said standard of proof; say because the defense manages to pinpoint contradictions in what the prosecution has presented. In such a context, the proposition *guilty or not* *guilty* is not understood as plainly true. Associated to it there is a proof of guiltiness; or a proof that this leads to contradictions. This intuitive understanding of *guilty or not guilty* departs from its classical interpretation and better fits in an intuitionistic setting.

Furthermore, as stated in [26], a proposition such as: *a verdict of guilty implies not innocent* is intuitively correct. The reason for this is that a verdict of *guilty*, as mentioned, is backed up by evidence meeting a standard of proof "beyond reasonable doubt", and such a proof can be used to convert any proof of *innocent* into a contradiction. But we might be more reluctant to accept the contraposition: *innocent implies not guilty* as intuitively correct. Being *innocent*, as a concept, is not backed up by any notion of evidence, neither it has to meet any standard of proof. A common starting point of a trial is the so-called principle of *presumption of innocence*, whereby someone *accused* of committing a crime is *a priori innocent*. In other words, some care needs to be taken in an intuitionistic setting for the law of contraposition does not necessarily hold.

The principle of *presumption of innocence* is clearly *defeasible*. If we only know that a person has been *accused of committing a crime*, we must conclude that this person is *innocent*. However, if additional information is brought up, e.g., a *credible witness*, the *murder weapon*, etc., the principle ceases to apply and the conclusion that the person is *innocent* is withdrawn. In other words, the principle of *presumption of innocence* behaves non-monotonically.

Here, we build a default logic over Intuitionistic Propositional Logic (IPL). Our aim is to formally reason about scenarios such as the one presented above. The choice of a default logic is not arbitrary. Since their introduction in [37], it has become clear that default logics have a special status in the literature on non-monotonic logic due to a relatively simple syntax and semantics, natural representation capabilities, and direct connections to other non-monotonic logics [2,5]. From a logic engineering point of view default logics are also interesting since they can be modularly built on an underlying logic having some minimal properties [8]. Moreover, we develop a tableaux calculus for our default logic taking some ideas from [7,9]. The resulting calculus is sound, complete, and terminating. Not many proof calculi for default logics built over IL exist. Important works are [1,14]. In [1], a tableaux method is presented for a default logic built on IPL which allows for the computation of extensions, but not for checking default consequence. The latter is covered in [14], where a sequent calculus is presented. The calculi introduced in [1,14] are related to ours but differ in some important aspects, in particular, in their construction. In addition, we present a prototype implementation which enables automated reasoning, a feature missing in [1,14].

Structure. In Sect. 2 we introduce preliminary definitions and results. More precisely, we present Intuitionistic Propositional Logic (IPL), and a default logic built over IPL ($\mathscr{D}$IPL). In Sect. 3 we recall tableaux for IPL and develop a tableaux calculus for default consequence in $\mathscr{D}$IPL. In Sect. 4 we present an implementation of our calculus. In Sect. 5 we report on a preliminary empirical evaluation of our implementation. In Sect. 6 we conclude the paper and discuss future research.

2 Basic Definitions

This section introduces basic definitions to make the paper self-contained.

Intuitionistic Logic. The syntax and semantics of Intuitionistic Propositional Logic (IPL) is defined below.

Definition 1 (Syntax). *The set $\mathscr{F}$ of wffs of* IPL *is defined on an enumerable set $\mathscr{P} = \{ p_i \mid 0 \leq i \}$ of* proposition symbols*, and is determined by the grammar*

$$\varphi ::= p_i \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \varphi \supset \varphi.$$

We write $\bot$ as an abbreviation for $p \wedge \neg p$, and $\top$ for $\neg\bot$.

As in [35], we define the semantics for IPL via intuitionistic Kripke models.

Definition 2 (Models). *A Kripke model $\mathfrak{M}$ is a tuple $\langle W, \preccurlyeq, V \rangle$ where: W is a non-empty set of elements (a.k.a.* worlds*); $\preccurlyeq\ \subseteq W \times W$ is the* accessibility relation*; and $V : W \to 2^{\mathscr{P}}$ is the* valuation function. *An* intuitionistic *Kripke model is a Kripke model $\mathfrak{M}$ in which $\preccurlyeq$ is* reflexive *and* transitive*, and in which V satisfies the so-called* heredity condition*: for all $w \preccurlyeq w'$, if $w \in V(p)$, $w' \in V(p)$.*

Definition 3 (Semantics). *Let $\mathfrak{M} = \langle W, \preccurlyeq, V \rangle$ be an intuitionistic Kripke model, $w \in W$, and $\varphi \in \mathscr{F}$, we define the* satisfiability relation $\mathfrak{M}, w \models \varphi$ *s.t.:*

$$\begin{array}{lll}
\mathfrak{M}, w \models p & \textit{iff} & p \in V(w) \\
\mathfrak{M}, w \models \varphi \wedge \psi & \textit{iff} & \mathfrak{M}, w \models \varphi \textit{ and } \mathfrak{M}, w \models \psi \\
\mathfrak{M}, w \models \varphi \vee \psi & \textit{iff} & \mathfrak{M}, w \models \varphi \textit{ or } \mathfrak{M}, w \models \psi \\
\mathfrak{M}, w \models \neg\varphi & \textit{iff} & \textit{for all } w \preccurlyeq w',\ \mathfrak{M}, w' \not\models \varphi \\
\mathfrak{M}, w \models \varphi \supset \psi & \textit{iff} & \textit{for all } w \preccurlyeq w',\ \textit{if } \mathfrak{M}, w' \models \varphi \textit{ then } \mathfrak{M}, w' \models \psi.
\end{array}$$

Notice that, unlike Classical Propositional Logic, $\mathfrak{M}, w \not\models \varphi$ is not equivalent to $\mathfrak{M}, w \models \neg\varphi$. For any $\Phi \subseteq \mathscr{F}$, we say that $\mathfrak{M}, w \models \Phi$ iff $\mathfrak{M}, w \models \varphi$, for all $\varphi \in \Phi$.

Next, we introduce the definition of *consequence* for IPL.

Definition 4 (Consequence). *Let $\Phi \subseteq \mathscr{F}$ and $\varphi \in \mathscr{F}$; we say that φ is a* logical consequence *of Φ, notation $\Phi \vDash \varphi$, iff for every $\mathfrak{M}$ and w in $\mathfrak{M}$, if $\mathfrak{M}, w \models \Phi$, then $\mathfrak{M}, w \models \varphi$*[1]*. We use $\vDash \varphi$ as an abbreviation for $\emptyset \vDash \varphi$. We say that Φ is consistent if $\Phi \not\vDash \bot$, otherwise it is inconsistent.*

It is well known that $\vDash$ satisfies *reflexivity*, *monotonicity*, *cut*, *structurality* and *compactness* (see e.g. [18] for details). In Definition 4 consequence in IPL is characterized semantically in terms of Kripke models. In Sect. 3 we present a syntactic characterization based on a proof system.

[1] This notion is referred to as *local consequence* in the literature on Modal Logic.

Default Logic. First introduced in [37], *Default Logic* comprises a sub-class of non-monotonic logics, characterized by so-called *defaults* and *extensions*. A default is a 3-tuple of formulas, notation $\pi \overset{\rho}{\Rightarrow} \chi$. Intuitively, we can think of a default as a *defeasible conditional* which given some conditions on π and ρ enables us to obtain χ. Extensions formalize what are these conditions. Defaults and extensions are introduced below.

Definition 5 (Defaults and Default Theories). *We call $\mathscr{D} = \mathscr{F}^3$ the set of all* defaults. *Let $\Delta \subseteq \mathscr{D}$, $\Delta^{\Pi} = \{ \pi \mid \pi \overset{\rho}{\Rightarrow} \chi \in \Delta \}$, $\Delta^{\mathrm{P}} = \{ \rho \mid \pi \overset{\rho}{\Rightarrow} \chi \in \Delta \}$ and $\Delta^{\mathrm{X}} = \{ \chi \mid \pi \overset{\rho}{\Rightarrow} \chi \in \Delta \}$. A default theory Θ is a pair (Φ, Δ) where $\Phi \subseteq \mathscr{F}$ and $\Delta \subseteq \mathscr{D}$. For any default theory $\Theta = (\Phi, \Delta)$, we define $\Phi_\Theta = \Phi$ and $\Delta_\Theta = \Delta$.*

In what follows we restrict our attention to finite default theories, i.e., default theories Θ in which both Φ_Θ and Δ_Θ are finite sets. Though our definitions extend directly to infinite default theories, there are some subtleties involved in dealing with infinite sets of defaults which we wish to avoid here (see [8] for details).

Definition 6 (Triggered). *Let Θ be a default theory, and $\Delta \cup \{\delta\} \subseteq \Delta_\Theta$; we say that δ is* triggered *by Δ iff $(\Phi_\Theta \cup \Delta^{\mathrm{X}}) \vDash \delta^{\Pi}$.*

Definition 7 (Blocked). *Let Θ be a default theory, and $\Delta \cup \{\delta\} \subseteq \Delta_\Theta$; we say that δ is* blocked *by Δ iff there is $\rho \in (\Delta \cup \delta)^{\mathrm{P}}$ s.t. $\Phi_\Theta \cup (\Delta \cup \delta)^{\mathrm{X}} \cup \rho$ is inconsistent.*

Definition 8 (Detached). *Let Θ be a default theory and $\Delta \cup \{\delta\} \subseteq \Delta_\Theta$; we say that δ is* detached *by Δ if δ is triggered and not blocked by Δ.*

Intuitively, for a default $\pi \overset{\rho}{\Rightarrow} \chi$ in the context of a default theory Θ, the notion of detachment in Definition 8 tells us under which conditions on π and ρ we can obtain χ. The definition of detachment is an intermediate step towards the definition of an extension.

Definition 9 (Generating Set). *Let Θ be a default theory and $\Delta \subseteq \Delta_\Theta$; we call Δ a* generating set *iff there is a total ordering $\lessdot$ on Δ_Θ s.t. $\Delta = \mathsf{D}^{\lessdot}_{\Theta}(n)$ for $n = |\Delta_\Theta|$, and $\mathsf{D}^{\lessdot}_{\Theta}$ is defined as:*

$$\mathsf{D}^{\lessdot}_{\Theta}(0) = \emptyset$$

$$\mathsf{D}^{\lessdot}_{\Theta}(i{+}1) = \begin{cases} \mathsf{D}^{\lessdot}_{\Theta}(i) \cup \delta & \textit{if } \delta \in \Delta_\Theta \backslash \mathsf{D}^{\lessdot}_{\Theta}(i) \textit{ is detached by } \mathsf{D}^{\lessdot}_{\Theta}(i), \textit{ and} \\ & \quad \textit{for all } \eta \neq \delta \in \Delta_\Theta \backslash \mathsf{D}^{\lessdot}_{\Theta}(i), \textit{ if } \eta \textit{ is detached by } \mathsf{D}^{\lessdot}_{\Theta}(i),\ \delta \lessdot \eta \\ \mathsf{D}^{\lessdot}_{\Theta}(i) & \textit{otherwise.} \end{cases}$$

Definition 10 (Extension). *Let Θ be a default theory, we say that $E \subseteq \mathscr{F}$ is an extension of Θ iff $E = \Phi_\Theta \cup \Delta^{\mathrm{X}}$ where $\Delta \subseteq \Delta_\Theta$ is a generating set.*

Intuitively, we can think of an extension of a default theory Θ as a set of formulas which contains Φ_Θ and which is closed under detachment. We are now in a position to define our default logic.

Definition 11 ($\mathscr{D}$IPL). *The default logic $\mathscr{D}$IPL is the 3-tuple $\langle \mathscr{F}, \vDash, \mathscr{E} \rangle$ where: (i) $\mathscr{F}$ is the set of all formulas of* IPL*; (ii) $\vDash$ is the consequence relation of* IPL*; and (iii) $\mathscr{E} : (2^{\mathscr{F}} \times 2^{\mathscr{D}}) \to 2^{(2^{\mathscr{F}})}$ is a function which maps every default theory Θ to its set of extensions, i.e., $E \in \mathscr{E}(\Theta)$ iff E is an* extension *of Θ (see Definition 10).*

The notion of consequence for $\mathscr{D}$IPL is introduced below.

Definition 12 (Default Consequence). *We say that a formula φ is a* default consequence *of a default theory Θ, notation $\Theta \approx\!\!\!\!\vDash \varphi$, iff for all $E \in \mathscr{E}(\Theta)$, $E \vDash \varphi$*[2]*.*

Some Comments and Easily Established Properties of $\mathscr{D}$IPL. Definition 10 corresponds to extensions as defined by Łukaszewicz in [27]. This definition of extensions is better behaved than Reiter's original proposal [37]. In particular, it guarantees existence, i.e., for any default theory Θ, $\mathscr{E}(\Theta) \neq \emptyset$. Definition 10 also guarantees *semi-monotonicity.* For default theories Θ_1 and Θ_2, define $\Theta_1 \sqsubseteq \Theta_2$ iff $\Phi_{\Theta_1} \subseteq \Phi_{\Theta_2}$ and $\Delta_{\Theta_1} \subseteq \Delta_{\Theta_2}$. Semi-monotonicity implies that for any two default theories $\Theta_1 \sqsubseteq \Theta_2$, if $\Phi_{\Theta_1} = \Phi_{\Theta_2}$, then for all $E_1 \in \mathscr{E}(\Theta_1)$, there is $E_2 \in \mathscr{E}(\Theta_2)$ s.t. $E_1 \subseteq E_2$. As we will see in Sect. 3, semi-monotonicity is important because it allows us to define a tableaux calculus for default consequence that can take advantage of a partial use of default theories. $\mathscr{D}$IPL is non-monotonic; in the sense that there are default theories Θ_1 and Θ_2 s.t. $\Theta_1 \sqsubseteq \Theta_2$, $\Theta_1 \approx\!\!\!\!\vDash \varphi$, and $\Theta_2 \not\approx\!\!\!\!\vDash \varphi$.

We conclude this section with an example illustrating some of the features and technical elements of $\mathscr{D}$IPL. The following terminology and notation is useful. A default $\pi \overset{\rho}{\Rightarrow} \chi$ is *normal* iff $\rho = \chi$. Normal defaults are written $\pi \Rightarrow \chi$. Let $\Theta_1 = (\Phi_1, \Delta_1)$ and $\Theta_2 = (\Phi_2, \Delta_2)$, define $\Theta_1 \sqcup \Theta_2 = (\Phi_1 \cup \Phi_2, \Delta_1 \cup \Delta_2)$. If Θ_1 is a default theory, Φ a set of formulas, and Δ a set of defaults, we use $\Theta_1 \sqcup \Phi$ to mean $\Theta \sqcup (\Phi, \emptyset)$, and $\Theta \sqcup \Delta$ to mean $\Theta \sqcup (\emptyset, \Delta)$.

Example 1 (Presumption of Innocence). Consider the following propositions:

> (1) 'accused'. (2) 'guilty or not guilty'. (3) 'guilty implies not innocent'. (4) 'the affidavit of a credible witness, the murder weapon, and the results of forensic tests, imply sufficient evidence'. (5) 'sufficient evidence implies a verdict of guilty'.

In IPL, we would typically formalize (1) to (5) as:

(1') a. (2') $g \vee \neg g$. (3') $g \supset \neg i$. (4') $(c \wedge w \wedge f) \supset e$. (5') $e \supset g$.

In turn, consider the principle of presumption of innocence, i.e., 'an accused of committing a crime is *a priori* innocent'; because of its defeasible status, we choose to formalize it as the (normal) default (6') $a \Rightarrow i$.

Let $\Theta = (\{g \vee \neg g, g \supset \neg i, (c \wedge w \wedge f) \supset e, e \supset g\}, \{a \Rightarrow i\})$; then:

[2] This notion is referred to as *sceptical consequence* in the literature on Default Logics.

(a) $\Theta \approx g \vee \neg g$ (e) $\Theta \sqcup \{a\} \approx a$ (h) $\Theta \sqcup \{a, c, w, f\} \approx a$
(b) $\Theta \approx g \supset \neg i$ (f) $\Theta \sqcup \{a\} \approx i$ (i) $\Theta \sqcup \{a, c, w, f\} \not\approx i$
(c) $\Theta \approx (c \wedge w \wedge f) \supset e$ (g) $\Theta \sqcup \{a\} \not\approx g$ (j) $\Theta \sqcup \{a, c, w, f\} \approx g$
(d) $\Theta \approx e \supset g$

Intuitively, the default theory Θ captures the basic set of assumptions discussed in the example in Sect. 1. These assumptions include the possible outcomes of the trial, i.e., the verdict of guilty or not guilty, i.e., $g \vee \neg g$; the consideration that guilty implies not innocent, i.e., $g \supset \neg i$; what constitutes sufficient evidence, i.e., $(c \wedge w \wedge f) \supset e$; and the claim that sufficient evidence leads to a verdict of guilty, i.e., $e \supset g$. Each of these assumptions is a default consequence of Θ. $\Theta \sqcup \{a\}$ considers the particular situation at the beginning of a trial, i.e., someone is accused of committing a crime. It follows that a and i are default consequences of $\Theta \sqcup \{a\}$; i.e., if the only thing we know is that someone is accused of committing a crime, we must conclude that said person is innocent, as per the principle of presumption of innocence. In turn, $\Theta \sqcup \{a, c, w, f\}$ captures the idea that if we acquire sufficient evidence, in the form of a credible witness affidavit, the murder weapon, and the results of forensic tests, we obtain a verdict of guilt, and in such a situation the principle of presumption of innocence no longer holds (i.e., no longer can be used as a basis for establishing the innocence of the accused).

3 Tableaux Proof Calculus

We develop a tableaux proof calculus for $\mathscr{D}$IPL based on one for IPL. The calculus captures default consequence in $\mathscr{D}$IPL and it is sound, complete, and terminating (using loop-checks).

Intuitionistic Tableaux. We begin by recalling the basics of a tableaux calculus for IPL. We follow closely the style of presentation of [35].

A *tableau* is a tree whose nodes are of two different kinds. The first kind corresponds to a pair of a formula φ and a natural number i, called a *label*, appearing in positive form, notation $@_i^+ \varphi$, or negative form, notation $@_i^- \varphi$[3]. The second kind corresponds to a pair of labels i and j, notation (i, j). Intuitively, $@_i^+ \varphi$ means "φ holds at world i"; and $@_i^- \varphi$ means "φ does not hold at world i"[4]. Intuitively, (i, j) means that world j is accessible from world i.

A tableau for φ is a tableau having $@_0^- \varphi$ as its root. A tableau for φ is *well-formed* if it is constructed according to the expansion rules in Fig. 1. In this figure, $(\wedge^+)$, $(\wedge^-)$, $(\vee^+)$ and $(\vee^-)$, are rules for the logical connectives of conjunction and disjunction. The "positive rules" $(\supset^+)$ and $(\neg^+)$ for the logical connectives of implication and negation are applied for every j in the branch, whereas the "negative rules" $(\supset^-)$ and $(\neg^-)$ for these logical connectives create

[3] The '@' notation is borrowed from Hybrid Logic [3].

[4] The signs $+$ and $-$ are necessary since in IPL we cannot use the symbol $\neg$ of negation for expressing that a formula does not hold in a world.

a “new” label j. The latter implies that the positive rules might need to be re-applied, e.g., if a new (i, j) is introduced in the branch. The rules (ref) and (trans) correspond to the reflexivity and transitivity constraints for the accessibility relation. The rule (her) corresponds to the heredity condition in Kripke models, propagating the valuation of a positive proposition symbol from a world to all its successors. The rule (A) occupies a special place in the construction of a tableau and will be discussed immediately below. Rules are applied as usual: premisses must belong to the branch; side conditions must be met (if any); the branch is extended at the level of leaves according to the consequents.

$$\dfrac{@_i^+(\varphi \wedge \psi)}{\begin{array}{c}@_i^+\psi\\ @_i^+\varphi\end{array}}\,(\wedge^+) \qquad \dfrac{@_i^-(\varphi \wedge \psi)}{@_i^-\varphi \quad @_i^-\psi}\,(\wedge^-) \qquad \dfrac{@_i^+(\varphi \vee \psi)}{@_i^+\varphi \quad @_i^+\psi}\,(\vee^+) \qquad \dfrac{@_i^-(\varphi \vee \psi)}{\begin{array}{c}@_i^-\psi\\ @_i^-\varphi\end{array}}\,(\vee^-)$$

$$\dfrac{\begin{array}{c}@_i^+(\varphi \supset \psi)\\ (i,j)\end{array}}{@_j^-\varphi \quad @_j^+\psi}\,(\supset^+) \qquad \dfrac{@_i^-(\varphi \supset \psi)}{\begin{array}{c}(i,j)\\ @_j^+\varphi\\ @_j^-\psi\end{array}}\,(\supset^-)^\dagger \qquad \dfrac{\begin{array}{c}@_i^+\neg\varphi\\ (i,j)\end{array}}{@_j^-\varphi}\,(\neg^+) \qquad \dfrac{@_i^-\neg\varphi}{\begin{array}{c}(i,j)\\ @_j^+\varphi\end{array}}\,(\neg^-)^\dagger$$

$$\dfrac{\begin{array}{c}@_i^+p\\ (i,j)\end{array}}{@_j^+p}\,(\mathsf{her})^\ddagger \qquad \dfrac{}{(i,i)}\,(\mathsf{ref})^* \qquad \dfrac{\begin{array}{c}(i,j)\\ (j,k)\end{array}}{(i,k)}\,(\mathsf{trans})^\P \qquad \dfrac{}{@_0^+\varphi}\,(\mathsf{A}) \quad \text{for } \varphi \in \Phi$$

† for j new (i.e., not used before in the branch).
‡ for $j \neq i$ in the branch.
* for i in the branch.
¶ for i, j, k in the branch.

Fig. 1. Tableau rules for IPL

Definition 13 (Closedness and Saturation). *A branch is* closed*, tagged* (▲)*, if* $@_i^+\varphi$ *and* $@_i^-\varphi$ *occur in the branch; otherwise it is* open*, tagged* (▼)*. A branch is* saturated*, tagged* (⧫)*, if the application of any expansion rule is redundant.*

Definition 14 (Provability). *A tableau* τ *for* φ *is an* attempt at proving φ*. We call* τ *a* proof of φ *if all branches in* τ *are closed. We write* $\vdash \varphi$ *if there is a proof of* φ*.*

Definition 13 introduces standard conditions of *closedness* and *saturation* for a tableau. Given these conditions, we define a tableau proof in Definition 14. The resulting proof calculus is sound and complete, i.e., $\vdash \varphi$ iff $\vDash \varphi$ (see [35]). Termination is ensured using loop-checks. Loop-checks are a standard termination technique in tableaux systems that require the re-application of expansion rules [17,23].

Tableaux constructed without the rule (A) formulate a *proof calculus* for *provability*, i.e., proofs without assumptions. Including the rule (A) in the construction of a tableau gives us a proof calculus for *deducibility* (proofs from a set

Φ of assumptions). Intuitively, (A) can be understood as stating that assumptions are always true in the "current" world. This rule is not strictly necessary: $\Phi \vDash \varphi$ iff $\vDash \bigwedge\Phi \supset \varphi$, for Φ finite. Nonetheless, incorporating a primitive rule for assumptions simplifies the definitions and understanding of tableaux for $\mathscr{D}$IPL. When (A) is involved, we talk about a tableau for φ from Φ. Such a tableau is *well-formed* if: its root is $@_0^- \varphi$; the rules in Fig. 1 are applied as usual; and (A) is applied w.r.t. the formulas in Φ. The precise definition of the proof calculus for deducibility is given in Definition 15. By adapting the argument presented in [35], it is possible to prove that the calculus for deducibility is also *sound* and *complete*, i.e., $\Phi \vdash \varphi$ iff $\Phi \vDash \varphi$. Termination is also guaranteed with loop-checks.

Definition 15 (Deducibility). *A tableau τ for φ from Φ is an* attempt at proving that φ *follows from Φ. We call τ a* proof of φ *from Φ if all branches in τ are closed. We write $\Phi \vdash \varphi$ if there is a proof of φ from Φ.*

An interesting feature of the tableaux calculus of Definition 15 is that not only it allows us to find proofs, but also lack of proofs. The latter is done by inspecting particular proof attempts, i.e., tableaux having a branch that is both open and saturated. These tableaux serve as counter-examples (i.e., they result in a description of a model which satisfies the assumptions but invalidates the formula we are trying to prove). This claim is made precise in Prop. 1. We resort to this feature as a way of *checking consistency of a set of formulas.* This check will be used in the definition of an expansion rule for tableaux for $\mathscr{D}$IPL.

Proposition 1. *If a tableau for φ from Φ has an open and saturated branch, then, $\Phi \nvDash \varphi$.*

Corollary 1. *$\Phi \nvdash \bot$ iff $\Phi \nvDash \bot$.*

Summing up, Definition 15 characterizes *proof theoretically*, via tableaux, the semantically defined notion of logical consequence in Definition 4. We repeat that termination of the proof calculus is not ensured by a simple exhaustive application of rules, and *loop-checks* are required. Intuitively, a loop-check restricts the application of an expansion rule ensuring that only "genuinely new worlds" are created. This technique, traced back to [17,23], is nowadays standard in tableaux systems.

Default Tableaux. *Default tableaux* extend tableaux for IPL with the addition of a new kind of node corresponding to the use of defaults. More precisely, a default tableau is a tree whose nodes are as in tableaux for IPL, together with a third kind that corresponds to the use of a default $\pi \overset{\rho}{\Rightarrow} \chi$. By a default tableau for φ from Θ, where φ is a formula and Θ is a default theory, we mean a default tableau having $@_0^- \varphi$ at its root. Such a default tableau is *well-formed* if it is constructed according to the expansion rules in Figs. 1 and 2. Rules in Fig. 1 are applied as before, with rule (A) being applied w.r.t. formulas in Φ_Θ. Rule (D) in Fig. 2 is applied w.r.t. the defaults in Δ_Θ. Note that rule (D) is applicable only if its side condition is met. This side condition can be syntactically decided using auxiliary tableaux in IPL to verify detachment, i.e., to prove that a default is triggered and not blocked.

$$\frac{}{\begin{array}{c}\delta_1\\ @_0^+\delta_1^{\mathrm{X}}\end{array} \qquad \dots \qquad \begin{array}{c}\delta_i\\ @_0^+\delta_i^{\mathrm{X}}\end{array} \qquad \dots \qquad \begin{array}{c}\delta_n\\ @_0^+\delta_n^{\mathrm{X}}\end{array}}\;(\mathsf{D})^{\dagger}$$

for $\{\,\delta_i \mid i \in [1,n]\,\} = \{\,\delta \in \Delta_\Theta \backslash \Delta_B \mid \delta$ is detached by $\Delta_B\,\}$
where Δ_B is the set of defaults in the branch.

Fig. 2. Tableau rule for defaults

Definition 16 (Default Deducibility). *Any well-formed default tableau τ for φ from Θ is an* attempt at proving that *φ follows from Θ by default. We call τ a* default proof *of φ from Θ if all branches of τ are closed. We write $\Theta \mathrel{|\!\!\sim} \varphi$ if there is a default proof of φ from Θ.*

The proof calculus just introduced is *sound* and *complete*. Termination is guaranteed by the termination of tableaux for IPL. Figure 3 shows a default proof.

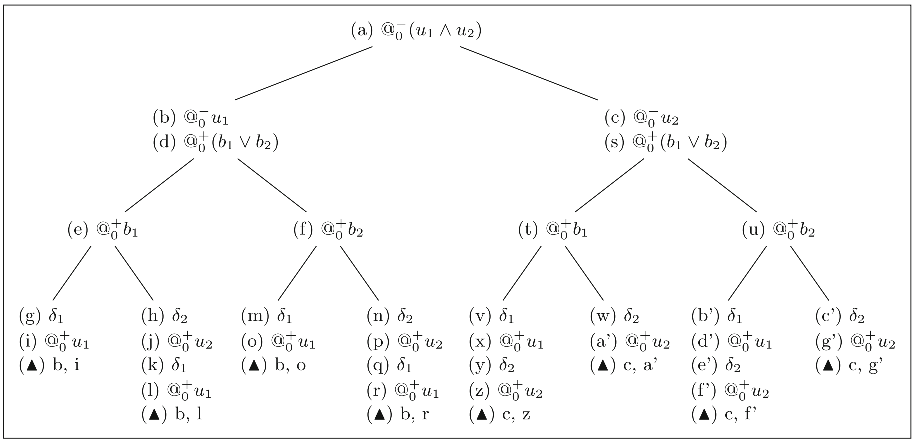

Fig. 3. Default tableau for $u_1 \wedge u_2$ from $\langle \{b_1 \vee b_2\}, \{\, \top \xrightarrow{u_i \wedge \neg b_i} u_i \mid i \in \{1,2\}\,\}\rangle$.

Theorem 1. *$\Theta \mathrel{|\!\!\sim} \varphi$ iff $\Theta \mathrel{|\!\!\approx} \varphi$. Since $\vdash$ terminates, $\mathrel{|\!\!\sim}$ also terminates.*

Proof (Sketch). We make use of the already known soundness and completeness of deducibility of the tableaux calculus for IPL. The proof of soundness and completeness of default deducibility depends on the following two observations: (i) a branch of a default tableau contains a branch of a tableau for deducibility in IPL (just remove default nodes); and (ii) defaults appearing in an open and saturated branch of a default tableau define a generating set (as per Definition 9) by construction. (i) and (ii) implies that if there is an open and saturated branch of a default tableau, then, there is an extension which serves as a counter-example. This proves that if $\Theta \approx \varphi$, then $\Theta \mathrel{|\!\sim} \varphi$. For the converse, suppose that $\Theta \not\approx \varphi$; hence there is an extension $E \in \mathscr{E}(\Theta)$ s.t. $E \not\models \varphi$. Let τ be any saturated default tableau for φ from Θ; by construction, there is an open branch of τ which contains a set of defaults which is a generating set for E. The result follows from the completeness of deducibility for IPL.

4 Implementation

Overview. DefTab is an implementation of the default tableau calculus presented in Sect. 3. It is available at http://tinyurl.com/deftab0.

Given Θ and φ as input, DefTab builds proof attempts of $\Theta \mathrel{|\!\sim} \varphi$ by searching for Kripke models for φ, and subsequently restricting these models with the use of sentences from Φ_Θ and defaults from Δ_Θ. DefTab reports whether or not a default proof has been found. In the latter case, DefTab exhibits an extension of Θ from which φ does not follow.

Defaults Detachment Rule and Sub-tableaux. The following is a brief explanation of how defaults are dealt with in DefTab. At any given moment, DefTab maintains defaults in three lists: *available*, *triggered*, and *detached*. The available list contains the defaults of the input default theory. An available default $\pi \overset{\rho}{\Rightarrow} \chi$ is triggered if π is deduced from the set Φ_Θ of the default theory Θ under question, together with the consequents of the defaults already in the branch of the tableau. Once triggered, available defaults are moved to the list of triggered defaults. DefTab uses the latter list to apply rule (D) in Fig. 2 and generates a (temporary) sub-list of non-blocked defaults. This sub-list of non-blocked defaults corresponds to the branching part of rule (D). DefTab then takes defaults from the sub-list of non-blocked defaults and moves them from the triggered list to the detached list, expanding the default tableau accordingly. Since the application of rule (D) requires checking consequence and consistency in IPL, for each (D)-step, DefTab builds a corresponding number of intuitionistic tableaux.

Blocking and Optimizations. For loop-checking intuitionistic rules that create new labels, DefTab uses a blocking technique called *pattern-based blocking* [25], initially designed for modal logics. In the present tableau system, when rule $(\neg^-)$ can be applied to some formula $@_i^- \neg\varphi$, it is first checked that no label k exists such that the set of formulas $\{@_k^- \varphi\} \cup @_k C(i)$ hold, where $C(i)$ are the constraints that formulas at label i forces on all its successors. If such a label exists, then the rule is not applied. The same occurs for rule $(\supset^-)$.

DefTab does not include semantic branching, or any other optimization based on Boolean negation and the excluded middle (which are usually unsound in an intuitionistic setup). Backjumping [24], on the other hand, is intuitionistically sound, and preliminary testing shows that it greatly improves performance.

We take special care of tracking dependencies of the consequent formulas introduced by the application of rule (D). That is, once a default $\pi \overset{\rho}{\Rightarrow} \chi$ is triggered, we bookkeep it along with the set of formulas that triggered it. Concretely, this bookkeeping is the union of the dependencies of all defaults Δ s.t. $\Phi_{\ominus} \cup \Delta^{X} \models \pi$. Note that this set can overestimate the set of dependencies of the triggered rule. This is a trade-off between being precise and limiting the number of sub-tableaux runs.

Usage. DefTab takes as input a file indicating the underlying logic to be used, a default theory partitioned into its set of formulas and set of defaults, and the formula to be checked. The structure of this input file is illustrated in the following example file `presumption.dt`. This file corresponds to Example 1 where: $g \mapsto$ P1, $i \mapsto$ P2, $a \mapsto$ P3, $c \mapsto$ P4, $w \mapsto$ P5, $f \mapsto$ P6, $e \mapsto$ P7.

```
intuitionistic
facts:
P1 v !P1; P1 -> !P2;
(P4 ^ P5 ^ P6) -> P7;
P7 -> P1; P3;
defaults:
P3 --> P2;
consequence:
P2
```

- The keyword `intuitionistic` indicates that the prover will work over IPL as the underlying logic.
- The keyword `facts` indicates the beginning of the set of formulas of the default theory.
- The keyword `defaults` indicates the beginning of the set of defaults. The syntax for a default $\pi \overset{\rho}{\Rightarrow} \chi$ is π `---` ρ `-->` χ. Normal default rules can be written as π `-->` χ.
- The keyword `consequence` indicates the formula to be proven.

DefTab is executed from the command line as

```
$ ./deftab -f presumption.dt
```

```
---------------------------------
Indeed a sceptical consequence.
Total time: 8.01513e-4
```

The output indicates that `P2` is a sceptical consequence of the default theory. If we add the facts `P4; P5; P6;`, we obtain that `P2` is not a sceptical consequence, indicated by the output: `Not a sceptical consequence, found bad extension: []`. The list `[]` indicates the defaults in the extension, in this case none. DefTab includes in the output a counter-example for disproving the candidate default consequence. Such counter-example consists of the generator set of the corresponding extension. In this case, the empty list indicates that the set of facts itself (without any default) is enough to generate the extension.

5 Preliminary Testing

To our knowledge, there is no standard test set for automated reasoning for default logic, and less so (if possible) for default reasoning based on IL. Moreover, our tool is a first prototype which still needs the implementation of many natural optimizations (e.g., catching for sub-tableaux results). Hence, any empirical testing is, by force, very preliminary, and should be taken only as a first evaluation of the initial performance of the tool, and of its current usability. Still, the results are encouraging and the prover seems to be able to handle examples which are well beyond what can be computed by hand, making it already a useful tool. We discuss below the tests sets we evaluated. All tests were performed on a machine running Ubuntu 16.04 LTS, with 8 GB of memory and an Intel Core i7-5500U CPU @ 2.40 Ghz.

Purely Intuitionistic Problems. When no defaults are specified in the input file, DefTab behaves as a prover for consequence in IPL (but it has not particular optimization for the case where the input is just an intuitionistic formula). Even though DefTab is still a prototype, we carried out a comparison of its performance w.r.t. existing provers for IPL.

We extend the comparison of provers for IPL carried out in [11] which compares their own prover (intuit) that implements satisfiability checking in IPL by an SMT (Satisfiability Modulo Theories) reasoner implemented over MiniSAT [13], with IntHistGC [21] and fCube [16]. These two last provers perform a backtracking search directly on a proof calculus, and are rather different from the approach taken by intuit, and closer to DefTab. IntHistGC implements clever backtracking optimizations that avoid recomputations in many cases. fCube implements several pruning techniques on a tableau-based calculus. Tests are drawn from three different benchmark suites.

1. ILTP [36] includes 12 problems parameterized by size. The original benchmark is limited, and hence it was extended as follows: two problems were generated up to size 38 and all other problems up to size 100, leading up to a total of 555 problem instances.
2. Benchmarks crafted by IntHistGC developers. These are 6 parameterized problems. They are carefully constructed sequences of formulas that separate classical and intuitionistic logic. The total number of instances is 610.
3. API solving; these are 10 problems where a rather large API (set of functions with types) is given, and the problem is to construct a new function of a given type. Each problem has variants with API sizes that vary in size from a dozen to a few thousand functions. These problems were constructed by the developers of intuit in an attempt to create practically useful problems. The total number of instances is 35.

Figure 4 shows a scatterplot (logscale) of the performance of DefTab with the other three provers, discriminating between valid and not valid formulas. Times shown are in seconds, and the timeout was set to 300 seconds. Point below the diagonal show cases where DefTab performance is worse that the other provers. The empirical tests show that specialized provers for IPL (and in particular intuit) outperforms DefTab, specially on valid, complex formulas. On simple, not valid formulas, the performance of DefTab in this test is better than fCube and comparable to IntHistGC.

Broken Arms [34]**.** Consider an assembly line with two mechanical arms. We assume that an arm is usable (u_i), but sometimes it can be broken (b_i), although a broken arm is an exception. In the literature on Default Logic such default assumptions have been formalized either as:

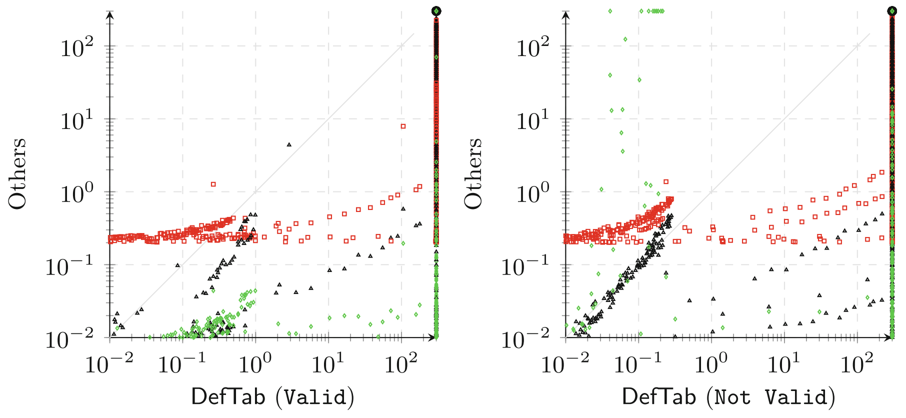

Fig. 4. Comparison on IPL: □ fCube, ◇ intuit, △ IntHistGC.

$$\Delta_1 = \left\{ \top \xRightarrow{u_i \wedge \neg b_i} u_i \;\middle|\; i \in \{1,2\} \right\} \text{ or } \Delta_2 = \left\{ (b_i \vee \neg b_i) \xRightarrow{u_i \wedge \neg b_i} u_i \;\middle|\; i \in \{1,2\} \right\}.$$

Δ_1 can be found in Poole's original discussion of the example (see [34]), whereas Δ_2 is found in, e.g., [1,14]. Suppose that we know as a fact that one of the arms is broken, but we do not know which one, i.e., $b_1 \vee b_2$. Let $\Theta_1 = (\{b_1 \vee b_2\}, \Delta_1)$ and $\Theta_2 = (\{b_1 \vee b_2\}, \Delta_2)$; it is possible to prove that $\Theta_1 \mathrel{|\!\!\sim} u_1 \wedge u_2$ and that $\Theta_2 \not\mathrel{|\!\!\sim} u_1 \wedge u_2$. Poole introduces this example to argue that having $u_1 \wedge u_2$ as a default consequence of Θ_1 is counter-intuitive; as we have as a fact that one of the arms is broken. This counter-intuitive result has inspired some important work on Default Logic [6,12,29]. Here, we choose this example, and Θ_1 in particular, merely as a test case. First, observe that Θ_1 can easily be made parametric on the number of arms (the number of defaults grows linearly with the number of arms). Second, observe that since defaults do not block each other, they can all be detached at any given time in a default proof attempt. The latter means that the prover needs, a priori, to consider a large number of combinations, leading to a potentially large search space. On the other hand, the intuitionistic reasoning needed is controlled, and as a result, the test case should mostly highlight how default are handled by the prover. These observations make this example a good candidate for testing the implementation of our proof calculus in order to evaluate its performance in a small, but non-trivial case. The results of running DefTab with n defaults are reported below.

No. of defaults	10	20	40	60	80
run time	0.038 s	0.499 s	8.667 s	49.748 s	177.204 s

Abstract. The following example, actually, an example template, is a variation of Broken Arms, where defaults do not block one another, but in which the detachment of defaults involves "non-trivial" intuitionistic formulas. This example is built on the ILTP library. To provide a bit of context, the ILTP library has two kinds of problems: those testing consequence in IPL, i.e., problems $\Phi \vdash \varphi$; and those testing not-consequence in IPL, i.e., problems $\Phi \nvdash \varphi$. Of the problems testing not-consequence, we are interested in a particular sub-class, i.e., those of the form $\Phi \nvdash \neg\varphi$. Problems in this sub-class can directly be used for testing blocking of defaults. Now, for each pair $\Phi_1 \vdash \varphi_1$ and $\Phi_2 \nvdash \neg\varphi_2$, we construct a default theory $\Theta = (\Phi_1 \cup \Phi_2, \{\varphi_1 \overset{\varphi_2}{\Longrightarrow} p\})$. We carry out this construction guaranteeing that the languages of $\Phi_i \cup \varphi_i$ are disjoint via renaming of proposition symbols and that p is new. Then, we consider a family $\{\, \Theta_i \mid i \in [1, n] \,\}$ of default theories constructed in the way just described. It can be proven that $(\bigsqcup_{i=1}^{n} \Theta_i) \mathrel{|\!\sim} (\bigwedge_{i=1}^{n} p_i)$. This construction serves as a way to test our implementation on increasingly more difficult default theories built on the ILTP library. The next table reports the results of running DefTab on $\Theta_n^m \mathrel{|\!\sim} (\bigwedge_{i=1}^{n} r_i)$ where:

$$\begin{aligned}
\Theta_b^a &= \textstyle\bigsqcup_{i=1}^{a}(\Phi_i^b \cup \Gamma_i^b, \{p_{i_0} \overset{q_{i_0}}{\Longrightarrow} r_i\}) \\
\Phi_b^a &= \{\quad p_{b_a}, \textstyle\bigwedge_{i=1}^{a}(p_{b_i} \supset (p_{b_i} \supset p_{b_{(i-1)}}))\} \\
\Gamma_b^a &= \{\neg\neg q_{b_a}, \textstyle\bigwedge_{i=1}^{a}(q_{b_i} \supset (q_{b_i} \supset q_{b_{(i-1)}}))\}.
\end{aligned}$$

$_n\backslash^m$	1	2	3	4	5
1	2.20e−3 s	1.50e−2 s	0.15 s	1.15 s	11.29 s
2	1.99e−3 s	2.23e−2 s	0.25 s	2.17 s	17.17 s
3	2.8e−3 s	3.7e−2 s	0.39 s	3.29 s	26.08 s

Intuitively, Θ_n^m contains m instances of sub-problems $\Phi_m^n \vdash p_{m_0}$ and $\Gamma_m^n \nvdash \neg q_{m_0}$ of size n taken from the ILTP. The sub-indices in Φ_m^n and Γ_m^n ensure languages are disjoint. It follows that every default $p_{i_0} \overset{q_{i_0}}{\Longrightarrow} r_i$ is detached by $\Phi_i^n \cup \Gamma_i^n$; and also that no default blocks any other. Thus, $\Theta_n^m \mathrel{|\!\sim} (\bigwedge_{i=1}^{m} r_i)$. The times show the progression of running DefTab on increasingly larger default theories Θ_n^m. Note the increase on complexity is both on m and n.

6 Final Remarks

We introduced $\mathscr{D}$IPL, a Default Logic to reason non-monotonically over Propositional Intuitionistic Logic. This logic is motivated by Normative Systems and Legal Artificial Intelligence in order to deal, e.g., with legal conflicts due to logical inconsistencies. Our contribution is twofold. First, we present a sound, complete and terminating tableaux calculus for $\mathscr{D}$IPL, which decides if a formula φ is a logical consequence of a default theory Θ. The calculus is based on tableaux for IPL as presented in [35], combined with the treatment for defaults of [7]. Second, we provide a prototype implementation for our calculus in the DefTab prover.

To the best of our knowledge, this is the first prover combining Intuitionistic Logic with Non-monotonic Reasoning. For instance, DeReS [10] is a default logic reasoner with an underlying propositional tableaux calculus. It is designed to check logical consequence by combining defaults reasoning and the underlying logic reasoning as 'black boxes'. This contrasts with DefTab which integrates these two reasonings in a same tableaux calculus. On the other hand DefTab only supports sceptical consequence checking, while DeReS also supports credulous consequence checking. In [15], a tableaux calculus for Intuitionistic Propositional Logic is presented, with a special treatment for nested implications. The implementation is no longer available, but it would be interesting to implement their specialized rules in DefTab. More directly related to our work are [1], where the authors present a sequent calculus for a Non-monotonic Intuitionistic Logic; and [14], where a tableaux method is presented but only for the computation of extensions, and with no implementation.

We ran an empirical evaluation of our prover and obtained preliminary results concerning the implementation. To test purely intuitionistic reasoning we used the ILTP problem library [36]. We tested non-monotonic features in two, very small, case examples (*broken arms* and *abstract*). Clearly further testing is necessary. In particular, we are interested in crafting examples that combine the complexities of non-monotonic and intuitionistic reasoning.

For future work there are several interesting lines of research. As we mentioned, the treatment of defaults in the calculus is almost independent from the underlying logic. As a consequence, it would be interesting to define a calculus which is parametric on the rules for the underlying logic (see, e.g., [8] for such a proposal). We believe that (modulo some refactoring in the current source code) the implementation of DefTab can be generalized to handle defaults over different underlying logics obtaining a prover for a wide family of Default Logics.

Ackowledgements. This work was partially supported by ANPCyT-PICTs-2017-1130 and 2016-0215, MinCyT Córdoba, SeCyT-UNC, the Laboratoire International Associé INFINIS and the European Union's Horizon 2020 research and innovation programme under the Marie Skodowska-Curie grant agreement No. 690974 for the project MIREL: MIning and REasoning with Legal texts.

References

1. Amati, G., Aiello, L., Gabbay, D., Pirri, F.: A proof theoretical approach to default reasoning I: tableaux for default logic. J. Log. Comput. **6**(2), 205–231 (1996)
2. Antoniou, G.: Nonmonotonic Reasoning. The MIT Press, Cambridge (1997)
3. Areces, C., ten Cate, B.: Hybrid logics. In: Blackburn, et al. [4], pp. 821–868
4. Blackburn, P., van Benthem, J., Wolter, F. (eds.): Handbook of Modal Logic. Elsevier, Amsterdam (2007)
5. Bochman, A.: Non-monotonic reasoning. In: Gabbay and Woods [20], pp. 555–632
6. Brewka, G.: Cumulative default logic: in defense of nonmonotonic inference rules. Artif. Intell. **50**(2), 183–205 (1991)

7. Cassano, V., Areces, C., Castro, P.: Reasoning about prescription and description using prioritized default rules. In: Barthe, G., Sutcliffe, G., Veanes, M. (eds.) 22nd International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR-22), EPiC Series in Computing, vol. 57, pp. 196–213. EasyChair (2018)
8. Cassano, V., Fervari, R., Areces, C., Castro, P.F.: Interpolation and beth definability in default logics. In: Calimeri, F., Leone, N., Manna, M. (eds.) JELIA 2019. LNCS (LNAI), vol. 11468, pp. 675–691. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-19570-0_44
9. Cassano, V., Pombo, C.G.L., Maibaum, T.S.E.: A propositional tableaux based proof calculus for reasoning with default rules. In: De Nivelle, H. (ed.) TABLEAUX 2015. LNCS (LNAI), vol. 9323, pp. 6–21. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24312-2_2
10. Cholewinski, P., Marek, V., Truszczynski, M.: Default reasoning system DeReS. In: 5th International Conference on Principles of Knowledge Representation and Reasoning (KR 1996), pp. 518–528. Morgan Kaufmann (1996)
11. Claessen, K., Rosén, D.: SAT modulo intuitionistic implications. In: Davis, M., Fehnker, A., McIver, A., Voronkov, A. (eds.) LPAR 2015. LNCS, vol. 9450, pp. 622–637. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48899-7_43
12. Delgrande, J., Schaub, T., Jackson, W.: Alternative approaches to default logic. Artif. Intell. **70**(1–2), 167–237 (1994)
13. Eén, N., Sörensson, N.: An extensible SAT-solver. In: Giunchiglia, E., Tacchella, A. (eds.) SAT 2003. LNCS, vol. 2919, pp. 502–518. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24605-3_37
14. Egly, U., Tompits, H.: A sequent calculus for intuitionistic default logic. In: 12th Workshop Logic Programming (WLP 1997), pp. 69–79 (1997)
15. Ferrari, M., Fiorentini, C., Fiorino, G.: A tableau calculus for propositional intuitionistic logic with a refined treatment of nested implications. J. Appl. Non-Class. Log. **19**, 149–166 (2009)
16. Ferrari, M., Fiorentini, C., Fiorino, G.: fCube: an efficient prover for intuitionistic propositional logic. In: Fermüller, C.G., Voronkov, A. (eds.) LPAR 2010. LNCS, vol. 6397, pp. 294–301. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16242-8_21
17. Fitting, M.: Proof Methods for Modal and Intuitionistic Logics. Springer, Dordrecht (1983). https://doi.org/10.1007/978-94-017-2794-5
18. Font, J.: Abstract Algebraic Logic: An Introductory Textbook, 1st edn. College Publications (2016)
19. Gabbay, D.M.: Intuitionistic basis for non-monotonic logic. In: Loveland, D.W. (ed.) CADE 1982. LNCS, vol. 138, pp. 260–273. Springer, Heidelberg (1982). https://doi.org/10.1007/BFb0000064
20. Gabbay, D., Woods, J. (eds.): Handbook of the History of Logic: The Many Valued and Nonmonotonic Turn in Logic, vol. 8. North-Holland, Amsterdam (2007)
21. 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 (LNAI), vol. 8562, pp. 262–268. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08587-6_19
22. Haeusler, E., de Paiva, V., Rademaker, A.: Intuitionistic logic and legal ontologies. In: 23rd International Conference on Legal Knowledge and Information Systems (JURIX 2010), Frontiers in Artificial Intelligence and Applications, vol. 223, pages 155–158. IOS Press (2010)

23. Hughes, G., Cresswell, M.: An Introduction to Modal Logic. Methuen, London (1968)
24. Hustadt, U., Schmidt, R.A.: Simplification and backjumping in modal tableau. In: de Swart, H. (ed.) TABLEAUX 1998. LNCS (LNAI), vol. 1397, pp. 187–201. Springer, Heidelberg (1998). https://doi.org/10.1007/3-540-69778-0_22
25. Kaminski, M., Smolka, G.: Terminating tableau systems for hybrid logic with difference and converse. J. Log. Lang. Inf. **18**(4), 437–464 (2009)
26. Kapsner, A.: The logic of guilt, innocence and legal discourse. In: Urbaniak, R., Payette, G. (eds.) Applications of Formal Philosophy. LARI, vol. 14, pp. 7–24. Springer, Cham (2017)
27. Łukaszewicz, W.: Considerations on default logic: an alternative approach. Comput. Intell. **4**, 1–16 (1988)
28. Makinson, D., van der Torre, L.: What is input/output logic? Input/output logic, constraints, permissions. In: Boella, G., van der Torre, L., Verhagen, H. (eds.) Normative Multi-agent Systems, Dagstuhl Seminar Proceedings, vol. 07122. Internationales Begegnungsund Forschungszentrum für Informatik (2007)
29. Mikitiuk, A., Truszczynski, M.: Constrained and rational default logics. In: 14th International Joint Conference on Artificial Intelligence (IJCAI 1995), pp. 1509–1517 (1995)
30. Osorio, M., Navarro Pérez, J., Arrazola, J.: Applications of intuitionistic logic in answer set programming. TPLP **4**(3), 325–354 (2004)
31. Parent, X., Gabbay, D., Torre, L.: Intuitionistic basis for input/output logic. In: Hansson, S.O. (ed.) David Makinson on Classical Methods for Non-Classical Problems. OCL, vol. 3, pp. 263–286. Springer, Dordrecht (2014). https://doi.org/10.1007/978-94-007-7759-0_13
32. Pearce, D.: Stable inference as intuitionistic validity. J. Log. Program. **38**(1), 79–91 (1999)
33. Pearce, D., Sarsakov, V., Schaub, T., Tompits, H., Woltran, S.: A polynomial translation of logic programs with nested expressions into disjunctive logic programs: preliminary report. In: Stuckey, P.J. (ed.) ICLP 2002. LNCS, vol. 2401, pp. 405–420. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45619-8_28
34. Poole, D.: What the lottery paradox tells us about default reasoning. In: 1st International Conference on Principles of Knowledge Representation and Reasoning (KR 1989), pp. 333–340 (1989)
35. Priest, G.: An Introduction to Non-Classical Logic: From If to Is. Cambridge University Press, Cambridge (2000)
36. Raths, T., Otten, J., Kreitz, C.: The ILTP problem library for intuitionistic logic. J. Autom. Reason. **38**(1–3), 261–271 (2007)
37. Reiter, R.: A logic for default reasoning. Artif. Intell. **13**(1–2), 81–132 (1980)
38. Servi, G.: Nonmonotonic consequence based on intuitionistic logic. J. Symb. Log. **57**(4), 1176–1197 (1992)
39. Wansing, H.: Semantics-based nonmonotonic inference. Notre Dame J. Formal Log. **36**(1), 44–54 (1995)