

A Tableau Decision Procedure for Propositional Intuitionistic Logic

Alessandro Avellone, Guido Fiorino, Ugo Moscato
Dipartimento di Metodi Quantitativi per l'Economia,
Università Milano-Bicocca, Piazza dell'Ateneo Nuovo, 1, 20126 Milano, Italy
{alessandro.avellone,guido.fiorino,ugo.moscato}@unimib.it

November 3, 2006

Abstract

The contribution of this paper consists in some techniques to bound the proof search space in propositional intuitionistic logic. These techniques are justified by Kripke semantics and they are the backbone of a tableau based theorem prover (PITP) implemented in C++ language. PITP and some known theorem provers are compared by the formulas of ILTP v1.1.1 benchmark library. It turns out that PITP is, at the moment, the propositional prover that solves most formulas of the library.

1 Introduction

The development of effective theorem provers for intuitionistic and constructive logics is of interest both for the investigations and applications of such logics to formal software/hardware verification and program synthesis (see i.e. [ML84, Con86, ES99, Men00, FFO02, BC04, AFF⁺06]).

In this paper we present a strategy and an implementation based on a tableau calculus for propositional intuitionistic logic. Our decision procedure implements the tableau calculus of [AFM04] (this calculus is an enhancement of the calculus given in [Fit69] and it is related to the tableau and sequent calculi of [MMO97, Hud93, Dyc92]).

We introduce some new techniques utilized by our decision procedure to narrow the search space and the width of the proofs. The PSPACE-completeness of intuitionistic validity ([Sta79]) suggests that backtracking and branching cannot be eliminated. In order to improve the time efficiency of the implementations and make them usable, strategies have to be developed to bound backtracking and branching as much as possible.

The optimizations we present are explained by the Kripke semantics for intuitionistic logic. Such semantical techniques are related to the fact that tableau calculi are strictly joined to the semantics of the logic at hand. Loosely speaking, a tableau proof for a formula is the attempt to build a model satisfying the formula. The construction of such a model proceeds by increasing, step by step, the information necessary to define such a model (thus, step by step the accuracy of the model increases). If the proof ends in a contradiction, then there is no model for the formula. Otherwise, a model satisfying

the formula is immediately derived from the proof. With this machinery at hand, first we provide a sufficient condition allowing us to stop a tableau proof without losing the completeness. Then we describe a technique to bound branching on the formulas which only contain conjunctions and disjunctions. Finally we present a technique to deduce the satisfiability of a set of formulas S , when the satisfiability of a set S' and a permutation τ such that $S = \tau(S')$ are known. Such a technique allows us to bound backtracking. Our technique to bound backtracking is different from the semantical technique provided in [Wei98].

Besides the strategy and its completeness, in the final part of the paper we present some experimental results on the implementation PITP. PITP is written in C++ and it is tested on the propositional part of ILTP v1.1.1 benchmark library ([ROK06]). Of 274 propositional benchmarks contained in ILTP v1.1.1, PITP decides 215 formulas including 13 previously unsolved problems within the time limit of ten minutes. To give the reader more elements to evaluate PITP strategies, comparisons with different versions of PITP are provided.

2 Notation and Preliminaries

We consider the propositional language \mathcal{L} based on a denumerable set of propositional variables or atoms \mathcal{PV} and the logical connectives $\neg, \wedge, \vee, \rightarrow$. (Propositional) Kripke models are the main tool to semantically characterize (propositional) intuitionistic logic **Int** (see [CZ97] and [Fit69] for the details). A Kripke model for \mathcal{L} is a structure $\underline{K} = \langle P, \leq, \rho, \Vdash \rangle$, where $\langle P, \leq, \rho \rangle$ is a poset with minimum ρ and \Vdash is the forcing relation, a binary relation between elements α of P and p of \mathcal{PV} such that $\alpha \Vdash p$ and $\alpha \leq \beta$ imply that $\beta \Vdash p$. The forcing relation is extended in a standard way to arbitrary formulas of \mathcal{L} as follows:

1. $\alpha \Vdash A \wedge B$ iff $\alpha \Vdash A$ and $\alpha \Vdash B$;
2. $\alpha \Vdash A \vee B$ iff $\alpha \Vdash A$ or $\alpha \Vdash B$;
3. $\alpha \Vdash A \rightarrow B$ iff, for every $\beta \in P$ such that $\alpha \leq \beta$, $\beta \Vdash A$ implies $\beta \Vdash B$;
4. $\alpha \Vdash \neg A$ iff for every $\beta \in P$ such that $\alpha \leq \beta$, $\beta \Vdash A$ does not hold.

We write $\alpha \not\Vdash A$ when $\alpha \Vdash A$ does not hold.

It is easy to prove that for every formula A , if $\alpha \Vdash A$ and $\alpha \leq \beta$, then $\beta \Vdash A$. A formula A is valid in a Kripke model $\underline{K} = \langle P, \leq, \rho, \Vdash \rangle$ if and only if $\rho \Vdash A$. It is well known that **Int** coincides with the set of formulas valid in all Kripke models.

If we consider Kripke models $\underline{K} = \langle P, \leq, \rho, \Vdash \rangle$ such that $|P| = 1$ we get classical models for (propositional) classical logic **Cl**. Classical models are usually seen as functions σ from \mathcal{PV} to $\{true, false\}$. Given a formula A and a model σ , we use $\sigma \models A$ with the usual meaning of satisfiability. Finally, given a set S , the set $\mathcal{PV}(S)$ denotes the elements of \mathcal{PV} occurring in S .

In the following presentation, we give a brief overview of the tableau calculus **Tab** of [AFM04] which is implemented by our decision procedure. The rules of the calculus are given in Table 1. The calculus works on signed well formed formulas (swff for short),

where a swff is a (propositional) formula prefixed with a sign \mathbf{T} , \mathbf{F} or \mathbf{F}_c . Given a Kripke model $\underline{K} = \langle P, \leq, \rho, \Vdash \rangle$, a world $\alpha \in P$, a formula A and a set of swffs S , the meaning of the signs is as follows:

- $\alpha \triangleright \mathbf{T}A$ (α realizes $\mathbf{T}A$) iff $\alpha \Vdash A$;
- $\alpha \triangleright \mathbf{F}A$ iff $\alpha \not\Vdash A$;
- $\alpha \triangleright \mathbf{F}_cA$ iff $\alpha \Vdash \neg A$;
- $\alpha \triangleright S$ iff α realizes every swff in S .
- $\underline{K} \triangleright S$ iff $\rho \triangleright S$.

A proof table (or proof tree) for S is a tree, rooted with S and obtained by the subsequent application of the rules of the calculus. As an example, let $\Gamma = \{\mathbf{T}(A \wedge B), \mathbf{T}(B \wedge C), \mathbf{F}(A \vee B)\}$. With “rule $\mathbf{T}\wedge$ applies to Γ taking $H \equiv \mathbf{T}(A \wedge B)$ as main swff” we mean that $\mathbf{T}\wedge$ applies to Γ as $\frac{\Gamma}{\Gamma \setminus \{\mathbf{T}(A \wedge B)\}, \mathbf{T}A, \mathbf{T}B} \mathbf{T}\wedge$. If no confusion arises we say that *a rule applies to Γ* or equivalently *a rule applies to H* . Finally with $Rule(H)$ we mean the rule related to H (in our example $Rule(\mathbf{T}(A \wedge B))$ is $\mathbf{T}\wedge$).

For every proof table for S , the depth and the number of symbols occurring in the nodes is linearly bounded in the number of symbols occurring in S (see [AFM04] for further details). This is the key feature to implement a depth-first decision procedure whose space complexity is $O(n \lg n)$ (as a matter of fact, it is well known that to generate all the proof tables in the search space and to visit them with a depth-first strategy, it is sufficient to have a stack containing, for every node of the visited branch, the index of the main swff and a bit to store if the leftmost branch has been visited [Hud93]).

We emphasize that the sign \mathbf{F}_c is introduced to give a specialized treatment of the negated formulas. In this sense the rules for the formulas signed with \mathbf{F}_c could be rewritten replacing in the \mathbf{F}_c -rules every occurrence of the sign \mathbf{F}_c with $\mathbf{T}\neg$.

Given a set of swffs S , the signed atoms of S are the elements of the set $\delta_S = \{H \mid H \in S \text{ and } H \text{ is a signed atom}\}$. We say that S contains a complementary pair iff $\{\mathbf{T}A, \mathbf{F}A\} \subseteq S$ or $\{\mathbf{T}A, \mathbf{F}_cA\} \subseteq S$. Given δ_S , we denote with σ_{δ_S} the (classical) model defined as follows: if $\mathbf{T}p \in \delta_S$, then $\sigma_{\delta_S}(p) = true$, $\sigma_{\delta_S}(p) = false$ otherwise.

Given a classical model σ , (I) $\sigma \triangleright \mathbf{T}H$ iff $\sigma \models H$; (II) $\sigma \triangleright \mathbf{F}H$ and $\sigma \triangleright \mathbf{F}_cH$ iff $\sigma \not\models H$. Given a set of swffs S , a swff H (respectively a set of swffs S') and the set of signed atoms δ_S defined as above, we say that δ_S realizes H (respectively δ_S realizes S'), and we write $\delta_S \triangleright H$ (respectively $\delta_S \triangleright S'$), iff there exists a classical model σ fulfilling the following conditions:

- (i) $\sigma \triangleright H$ (respectively $\sigma \triangleright H'$ for every $H' \in S'$).
- (ii) if $\mathbf{T}p \in \delta_S$, then $\sigma(p) = true$;
- (iii) if $\mathbf{F}p \in \delta_S$ or $\mathbf{F}_cp \in \delta_S$, then $\sigma(p) = false$

In other words, the relation \triangleright between δ_S and a signed formula H holds if there exists a classical model σ both realizing δ_S and H . If $\{\mathbf{T}p, \mathbf{F}_cp, \mathbf{F}p\} \cap S = \emptyset$ then there is no condition on the truth value that σ gives to p .

$\frac{S, \mathbf{T}(A \wedge B)}{S, \mathbf{TA}, \mathbf{TB}} \mathbf{T}^\wedge$	$\frac{S, \mathbf{F}(A \wedge B)}{S, \mathbf{FA} S, \mathbf{FB}} \mathbf{F}^\wedge$	$\frac{S, \mathbf{F}_c(A \wedge B)}{S_c, \mathbf{F}_cA S_c, \mathbf{F}_cB} \mathbf{F}_c^\wedge$
$\frac{S, \mathbf{T}(A \vee B)}{S, \mathbf{TA} S, \mathbf{TB}} \mathbf{T}^\vee$	$\frac{S, \mathbf{F}(A \vee B)}{S, \mathbf{FA}, \mathbf{FB}} \mathbf{F}^\vee$	$\frac{S, \mathbf{F}_c(A \vee B)}{S, \mathbf{F}_cA, \mathbf{F}_cB} \mathbf{F}_c^\vee$
$\frac{S, \mathbf{TA}, \mathbf{T}(A \rightarrow B)}{S, \mathbf{TA}, \mathbf{TB}} \mathbf{T} \rightarrow \text{Atom},$ with A an atom		
$\frac{S, \mathbf{F}(A \rightarrow B)}{S_c, \mathbf{TA}, \mathbf{FB}} \mathbf{F} \rightarrow \quad \frac{S, \mathbf{F}_c(A \rightarrow B)}{S_c, \mathbf{TA}, \mathbf{F}_cB} \mathbf{F}_c \rightarrow$		
$\frac{S, \mathbf{T}(\neg A)}{S, \mathbf{F}_cA} \mathbf{T}^\neg$	$\frac{S, \mathbf{F}(\neg A)}{S_c, \mathbf{TA}} \mathbf{F}^\neg$	$\frac{S, \mathbf{F}_c(\neg A)}{S_c, \mathbf{TA}} \mathbf{F}_c^\neg$
$\frac{S, \mathbf{T}((A \wedge B) \rightarrow C)}{S, \mathbf{T}(A \rightarrow (B \rightarrow C))} \mathbf{T} \rightarrow \wedge \quad \frac{S, \mathbf{T}(\neg A \rightarrow B)}{S_c, \mathbf{TA} S, \mathbf{TB}} \mathbf{T} \rightarrow \neg$		
$\frac{S, \mathbf{T}((A \vee B) \rightarrow C)}{S, \mathbf{T}(A \rightarrow p), \mathbf{T}(B \rightarrow p), \mathbf{T}(p \rightarrow C)} \mathbf{T} \rightarrow \vee$		
$\frac{S, \mathbf{T}((A \rightarrow B) \rightarrow C)}{S_c, \mathbf{TA}, \mathbf{F}p, \mathbf{T}(p \rightarrow C), \mathbf{T}(B \rightarrow p) S, \mathbf{TC}} \mathbf{T} \rightarrow \rightarrow$		
where $S_c = \{\mathbf{TA} \mathbf{TA} \in S\} \cup \{\mathbf{F}_cA \mathbf{F}_cA \in S\}$ and p is a new atom		

Table 1: the **Tab** calculus

We say that a wff or a swff is classically evaluable (*cle*-wff and *cle*-swff for short) iff conjunctions and disjunctions are the only connectives occurring in it. Finally, a set S of swffs is contradictory if at least one of the following conditions holds:

1. S contains a complementary pair;
2. S contains a *cle*-swff H such that $\delta_S \not\vdash H$;
3. $\delta_S \not\vdash S$ and for every propositional variable p occurring in S , $\mathbf{T}p \in S$ or $\mathbf{F}_cp \in S$.

Proposition 2.1 *If a set of swffs S is contradictory, then for every Kripke model $\underline{K} = \langle P, \leq, \rho, \Vdash \rangle$, $\rho \not\vdash S$.*

Proof: If S is contradictory because the first condition holds, then for some formula A , $\{\mathbf{TA}, \mathbf{FA}\} \subseteq S$ or $\{\mathbf{TA}, \mathbf{F}_cA\} \subseteq S$ holds. By the meaning of the signs and the definition of the forcing relation in Kripke models, the claim immediately follows. If S is contradictory because the second condition holds, then $\delta_S \not\vdash H$. Thus, there is no classical model realizing both the signed atoms of S (that is δ_S) and H . Since H

is a *cle*-swff, its classical and intuitionistic realizability coincide, thus no Kripke model realizes S . If S is contradictory because the third condition holds, then let us suppose there exists a Kripke model $\underline{K} = \langle P, \leq, \rho, \Vdash \rangle$ such that $\rho \triangleright S$. Then for every $p \in \mathcal{PV}(S)$, $\rho \Vdash p$ or $\rho \Vdash \neg p$ and this means that every world in \underline{K} forces the same propositional variables occurring in S , that is for every $\alpha, \beta \in P$ and for every $p \in \mathcal{PV}(S)$, $\alpha \Vdash p$ iff $\beta \Vdash p$. Let $\phi \in P$ be a maximal state in the poset $\langle P, \leq, \rho \rangle$. Since ϕ behaves as a classical model, by hypothesis, $\phi \not\triangleright S$. Then, since every world of \underline{K} forces the same propositional variables of S we deduce that $\rho \not\triangleright S$. \square

A closed proof table is a proof table whose leaves are all contradictory sets. A closed proof table is a proof of the calculus: a formula A is provable if there exists a closed proof table for $\{\mathbf{F}A\}$. For every rule of the calculus it is easy to prove that if there exists a Kripke model $\underline{K} = \langle P, \leq, \rho, \Vdash \rangle$ and $\alpha \in P$ such that α realizes the premise of the rule, then there exists (a possibly different) Kripke model $\underline{K}' = \langle P', \leq', \rho', \Vdash' \rangle$ and $\beta \in P'$ such that β realizes the conclusion. This is the main step to prove the soundness of the calculus:

Theorem 2.2 (Soundness) *Let A be a wff. If there exists a closed proof table starting from $\{\mathbf{F}A\}$, then A is valid.*

3 The optimizations and the proof search algorithm

Below we describe a recursive procedure $\text{TAB}(S)$ which given a set S of swffs, returns either a closed proof table for S or NULL (if there exists a Kripke model realizing S).

To describe TAB we use the following notation. Let S be a set of swffs, let $H \in S$ and let S_1 or $S_1 \mid S_2$ be the nodes of the proof tree obtained by applying to S the rule $\text{Rule}(H)$ corresponding to H . If Tab_1 and Tab_2 are closed proof tables for S_1 and S_2 respectively, then $\frac{S}{\text{Tab}_1} \text{Rule}(H)$ or $\frac{S}{\text{Tab}_1 \mid \text{Tab}_2} \text{Rule}(H)$ denote the closed proof table for S defined in the obvious way. Moreover, $\mathcal{R}_i(H)$ ($i = 1, 2$) denotes the set containing the swffs of S_i which replaces H . For instance:

$$\begin{aligned} \mathcal{R}_1(\mathbf{T}(A \wedge B)) &= \{\mathbf{T}A, \mathbf{T}B\}, \\ \mathcal{R}_1(\mathbf{T}(A \vee B)) &= \{\mathbf{T}A\}, \mathcal{R}_2(\mathbf{T}(A \vee B)) = \{\mathbf{T}B\}, \\ \mathcal{R}_1(\mathbf{T}((A \rightarrow B) \rightarrow C)) &= \{\mathbf{T}A, \mathbf{F}p, \mathbf{T}(B \rightarrow p), \mathbf{T}(p \rightarrow C)\}, \\ \mathcal{R}_2(\mathbf{T}((A \rightarrow B) \rightarrow C)) &= \{\mathbf{T}C\}. \end{aligned}$$

As stated in the introduction, TAB uses substitutions. For our purposes, the only substitutions we are interested in are permutations from \mathcal{PV} to \mathcal{PV} . Given a substitution τ and a swff H (respectively, a set of swffs S and tableau T) $\tau(H)$ (respectively, $\tau(S)$ and $\tau(T)$) means the swff (respectively, the set of swffs and the tableau) obtained by applying the substitution in the obvious way.

The procedure TAB divides the formulas in six groups according to their behavior with respect to branching and backtracking:

$$\begin{aligned} \mathcal{C}_1 &= \{\mathbf{T}(A \wedge B), \mathbf{F}(A \vee B), \mathbf{F}_c(A \vee B), \mathbf{T}(\neg A), \mathbf{T}(p \rightarrow A) \text{ with } p \text{ an atom}, \mathbf{T}((A \wedge B) \rightarrow C), \mathbf{T}((A \vee B) \rightarrow C)\}; \\ \mathcal{C}_2 &= \{\mathbf{T}(A \vee B), \mathbf{F}(A \wedge B)\}; \\ \mathcal{C}_3 &= \{\mathbf{F}(\neg A), \mathbf{F}(A \rightarrow B)\}; \end{aligned}$$

$$\mathcal{C}_4 = \{\mathbf{T}((A \rightarrow B) \rightarrow C), \mathbf{T}(\neg A \rightarrow B)\};$$

$$\mathcal{C}_5 = \{\mathbf{F}_c(A \rightarrow B), \mathbf{F}_c(\neg A)\};$$

$$\mathcal{C}_6 = \{\mathbf{F}_c(A \wedge B)\}.$$

We call \mathcal{C}_i -swffs ($i = 1, \dots, 6$), respectively \mathcal{C}_i -rules, the swffs of the group \mathcal{C}_i , respectively the rules related to \mathcal{C}_i -swffs.

The intuition behind these groups can be explained as follows. It is well known that in classical logic the order in which the rules are applied does not affect the completeness of the decision procedure: if a proof exists it is found independently of the order in which the rules are applied. Thus the search space consists of a single proof tree whose branches have to be visited by the decision procedure. The rules having two conclusions give rise to branches. Rules of this kind are $\mathbf{T}\vee$ and $\mathbf{F}\wedge$. In intuitionistic logic the order in which the rules are applied is relevant and affect the completeness. Given a set Γ , there are many ways to go on with the proof (that is many swffs can be chosen as main swff). Since the order is relevant, if the choice of a swff as main swff does not give a closed proof table, we have to backtrack and try with another swff as main swff. This means that in intuitionistic logic there is a space of proof tables to be visited by backtracking. Rules requiring backtracking are, i.e., $\mathbf{F}\rightarrow$, $\mathbf{T}\rightarrow\rightarrow$. In order to bound time consumption, TAB applies the rules not requiring branching and backtracking first, then the rules not requiring backtracking, finally the rules requiring backtracking. In the Completeness Lemma (Lemma 4.1, page 11) we prove that we do not lose completeness if \mathcal{C}_5 and \mathcal{C}_6 -rules are applied only when no other rule applies. Thus the application of \mathcal{C}_5 and \mathcal{C}_6 -rules is invertible and no backtracking is required. On the other hand, to get completeness, backtracking is unavoidable when \mathcal{C}_3 and \mathcal{C}_4 -rules are applied.

Now we come to the optimizations. First we discuss two checks that allow us to bound the depth of the proofs. Let S be a set such that $\sigma_{\delta_S} \triangleright S$. Thus the Kripke model coinciding with the classical model σ_{δ_S} realizes S and we do not need to go on with the proof. The second check is related to Point 3 in the definition of contradictory set. If $\delta_S \not\triangleright S$ and every propositional variable occurring in S occurs in S as swff signed \mathbf{T} or \mathbf{F}_c , then there is no Kripke model realizing S and we do not need to proceed with the proof. Although these checks could be performed after every rule application, our strategy performs it when neither a \mathcal{C}_1 nor a \mathcal{C}_2 -rule applies to S (in Section 5 this optimization is referred as **opt1**).

In order to bound branching, TAB treats in a particular way the *cle*-swffs in S , that is the swffs in which only \wedge and \vee occur (in other words swffs whose intuitionistic truth coincides with classical truth). When TAB treats \mathcal{C}_2 -swffs (Point 3 of the algorithm given below), first of all TAB checks if in S there exists a *cle*-swff H such that $\delta_S \not\triangleright H$. If S fulfills this condition, then S is not realizable, as we pointed out in Proposition 2.1. Otherwise, if in S a *cle*-swff H occurs, such that σ_{δ_S} does not satisfy H , the splitting rule $Rule(H)$ is applied (we recall that σ_{δ_S} is the classical model defined from S by taking as true the atoms p such that $\mathbf{T}p \in S$). Thus the *cle*-swffs of S , satisfying the underlying model, are never applied. Despite this, from the realizability of one among $(S \setminus \{H\}) \cup \mathcal{R}_1(H)$ and $(S \setminus \{H\}) \cup \mathcal{R}_2(H)$ we have enough information to prove the realizability of S . In other words, we consider only the *cle*-swffs of \mathcal{C}_2 that are not realized by the model underlying S (see Point 3 and Completeness Lemma for the

details). As an example consider

$$S = \{ \mathbf{F}(P0 \wedge P2), \mathbf{F}(P0 \wedge P4), \mathbf{F}(P2 \wedge P4), \mathbf{F}(P1 \wedge P3), \mathbf{F}(P1 \wedge P5), \\ \mathbf{F}(P3 \wedge P5), \mathbf{T}(P0 \vee P1), \mathbf{T}(P2 \vee P3), \mathbf{T}(P4 \vee P5) \}.$$

Since σ_{δ_S} realizes the \mathbf{F} -swffs in S but σ_{δ_S} does not realize any of the \mathbf{T} -swffs of S , then TAB chooses one of them, let us suppose $H = \mathbf{T}(P0 \vee P1)$. The rule $\mathbf{T}\vee$ is applied to S and $S_1 = (S \setminus \{H\}) \cup \{\mathbf{TP0}\}$ and $S_2 = (S \setminus \{H\}) \cup \{\mathbf{TP1}\}$ are the subsequent sets of S . Now consider S_1 . Since $\sigma_{\delta_{S_1}}$ realizes $\mathbf{TP0}$ and all the \mathbf{F} -swffs in S_1 , but $\sigma_{\delta_{S_1}}$ realizes neither $\mathbf{T}(P2 \vee P3)$ nor $\mathbf{T}(P4 \vee P5)$, TAB chooses one of them, let us suppose $H = \mathbf{T}(P2 \vee P3)$. The rule $\mathbf{T}\vee$ is applied to S_1 and $S_3 = (S_1 \setminus \{H\}) \cup \{\mathbf{TP2}\}$ and $S_4 = (S_1 \setminus \{H\}) \cup \{\mathbf{TP3}\}$ are the subsequent sets of S_1 . Since δ_{S_3} does not realize $\mathbf{F}(P0 \wedge P2)$ we deduce that S_3 is contradictory. Similarly for the other sets. We emphasize that at every step, the \mathcal{C}_2 -rules applicable to S_i are only the ones where the related swffs are not realized by $\sigma_{\delta_{S_i}}$. Without this strategy a huge closed proof table could arise (in Section 5 this optimization is referred as **opt2**).

To bound the search space, [Wei98] describes a decision procedure in which backtracking is bounded by a semantical technique inspired by the completeness theorem. The completeness theorem proves the satisfiability (realizability in our context) of a set S under the hypothesis that S does not have any closed proof table. As an example, let $S = \{\mathbf{F}(A \rightarrow B), \mathbf{T}((A \rightarrow B) \rightarrow C), \mathbf{FC}, \mathbf{TD}\}$. From S we can define the Kripke model $\underline{K}(S) = \langle P, \leq, \rho, \Vdash \rangle$ such that $P = \{\rho\}$ and $\rho \Vdash D$. Note that $\underline{K}(S)$ realizes \mathbf{TD} but $\underline{K}(S)$ does not realize S . To prove the realizability of S , the realizability of $S_1 = \{\mathbf{TA}, \mathbf{FB}, \mathbf{T}((A \rightarrow B) \rightarrow C), \mathbf{TD}\}$ and one between $S_2 = \{\mathbf{TA}, \mathbf{Fp}, \mathbf{T}(B \rightarrow p), \mathbf{T}(p \rightarrow C), \mathbf{TD}\}$ and $S_3 = \{\mathbf{F}(A \rightarrow B), \mathbf{TC}, \mathbf{FC}, \mathbf{TD}\}$ have to be proved. Since S_3 is not realizable, the realizability of S_1 and S_2 must be proved. From S_1 we define the Kripke model $\underline{K}(S_1) = \langle \{\alpha\}, \leq, \alpha, \Vdash \rangle$, where $\alpha \leq \rho$, $\alpha \Vdash D$ and $\alpha \Vdash A$ such that $\underline{K}(S_1) \triangleright S_1$. If we glue $\underline{K}(S_1)$ above $\underline{K}(S)$ we get a new Kripke model $\underline{K} = \langle \{\rho, \alpha\}, \leq, \rho, \Vdash \rangle$ where $\rho \leq \rho$, $\rho \leq \alpha$, $\alpha \leq \alpha$, $\rho \Vdash D, \alpha \Vdash D$ and $\alpha \Vdash A$. Since $\underline{K} \triangleright S$, we do not need to apply $\mathbf{T} \rightarrow \rightarrow$ to S in order to obtain $S_2 = \{\mathbf{TA}, \mathbf{Fp}, \mathbf{T}(B \rightarrow p), \mathbf{T}(B \rightarrow C), \mathbf{TD}\}$ (from S_2 a Kripke model $\underline{K}(S_2)$ is definable; $\underline{K}(S_2)$ glued above $\underline{K}(S)$ gives rise to a Kripke model realizing S). In this case the work on S_2 is spared. In the general case, see [Wei98], the information collected from non closed proof tables built from a set S is used to build a Kripke model \underline{K} . As a matter of fact, let S be a set such that no \mathcal{C}_1 or \mathcal{C}_2 -rule is applicable. Let $\{H_1, \dots, H_u\} \subseteq S$ the \mathcal{C}_3 and \mathcal{C}_4 -swffs of S . If there exists a H_j such that $\underline{K} \not\triangleright H_j$, then $\text{Rule}(H_j)$ have to be applied. If a closed proof table is found, then S is not realizable, otherwise the Kripke model \underline{K} can be extended in a new one, \underline{K}_j satisfying H_j . The procedure of [Wei98] continues until a closed proof table or a Kripke model \underline{K}_i , $1 \leq i \leq u$, such that $\underline{K}_i \triangleright \{H_1, \dots, H_u\}$ is found. The procedure prunes the search space, since in S not all the swffs requiring backtracking are considered, but only the swffs which, when checked, are not realized from the Kripke model at hand.

Now, consider $S = \{\mathbf{F}(A \rightarrow B), \mathbf{F}(C \rightarrow D)\}$. From S we can define the Kripke model $\underline{K}(S) = \langle P, \leq, \rho, \Vdash \rangle$ such that $P = \{\rho\}$ and \Vdash is the empty set. $\underline{K}(S)$ does not realize S . By applying $\mathbf{F} \rightarrow$ to S with $\mathbf{F}(A \rightarrow B)$ as the main formula we get $S_1 = \{\mathbf{TA}, \mathbf{FB}\}$. The underlying model is $\underline{K}(S_1) = \langle \{\alpha\}, \leq, \alpha, \Vdash \rangle$ with $\alpha \Vdash A$. $\underline{K}(S_1)$ glued above $\underline{K}(S)$ gives rise to a model that does not realize $\mathbf{F}(C \rightarrow D)$. Thus we must backtrack. We

provable				
	0-1s	1-10s	10-200s	200-300s
STRIP	200	19	15	0
PITP	192	36	11	2
unprovable				
	0-1s	1-10s	10-200s	200-300s
STRIP	10139	404	394	49
PITP	11663	49	27	4
	> 300s			
STRIP	780			
PITP	16			

Table 5: PITP and STRIP on random generated formulas

- [Sta79] R. Statman. Intuitionistic logic is polynomial-space complete. *Theoretical Computer Science*, 9(1):67–72, 1979.
- [Wei98] Klaus Weich. Decision procedures for intuitionistic propositional logic by program extraction. In *TABLEAUX*, pages 292–306, 1998.