Decision Procedures for Intuitionistic Propositional Logic by Program Extraction

Klaus Weich

Mathematisches Institut der Universität München, Theresienstr. 39,
D–80333 München, Germany; tel: +49 89 2394 4417;
email: weich@rz.mathematik.uni-muenchen.de

Abstract. We present two constructive proofs of the decidability of intuitionistic propositional logic by simultaneously constructing either a counter–model or a derivation. From these proofs, we extract two programs which have a sequent as input and return a derivation or a counter–model. The search tree of these algorithms is linearly bounded by the number of connectives of the input. Soundness of these programs follows from giving a correct construction of the derivations, similarly to Hudelmaier's work [7]; completeness from giving a correct construction of the counter–models, inspired by Miglioli, Moscato, and Ornaghi [8].

1 Introduction

Intuitionistic proofs can be considered as programs together with their verification. Consequently intuitionistic logic is a method for developing correct programs. To demonstrate the advantage of this approach, we construct two theorem provers for the propositional part by extracting them from a decidability proof.

Taking up Fitting's [2] completeness proof, Underwood [12] outlined how a decidability proof could be implemented in a formal system like NUPRL. For this, she proved that each set of sequents either contains a provable one or has a Kripke model so that each sequent is refuted at a certain node. The sequents themselves correspond to the nodes of a tableau. The proof is by induction on the number of formulas which can be added to the set of sequents, corresponding to a tableau rule. Underwood's extracted program has a sequent as input and returns a derivation or a counter–model. Her program uses a loop–check and tries to construct bottom–up a repetition–free derivation in a sequent calculus, similarly to the decision procedure already given by Gentzen [3].

To avoid a loop–check, contraction–free sequent calculi were introduced by Hudelmaier [6] and Dyckhoff [1], rediscovering the work of Vorob'ev [13]. The main idea of their completeness proof is that every derivation in a sequent calculus can be transformed so that every left premise of a left rule for implication (L-$\supset$ in our notation below) is either an axiom or the conclusion of a right rule (R-...). Formalising their proof would be a thankless and hard task. Fortunately, Miglioli, Moscato, and Ornaghi [8] gave an alternative completeness proof of a similar system by constructing a Kripke model. Hudelmaier developed his calculus further and gave an $O(n \log n)$–space algorithm [7].

H. de Swart (Ed.): TABLEAUX'98, LNAI 1397, pp. 292–306, 1998.

In Sect. 4, we give a new proof of the decidability of the intuitionistic propositional logic. For this, we show that for every sequent there is a derivation or a counter–model. We simultaneously construct a derivation, taking up Hudelmaier's approach [7], or a counter–model, simplifying Miglioli, Moscato, and Ornaghi's idea [8]. The extracted algorithm has a sequent as input and returns a derivation or a counter–model. The height of the search tree of this algorithm is linearly bounded by the number of logical connectives of the input. Indeed, our algorithm restricted to derivability coincides with Hudelmaier's [7]. In addition, our algorithm terminates faster than the algorithms for refutability presented by Pinto and Dyckhoff [9] and Hudelmaier [4, 5], the search tree of which is only exponentially bounded.

In Sect. 5, we describe how the counter–models constructed during the search can be used to prune the search space.

In Sect. 6, we present an alternative proof of decidability. The algorithm described by this proof is completely different. Whereas the algorithm of Sect. 4 examines the premise A of implications $A \supset B$ in the left–hand side of the sequents, the one of Sect. 6 looks at the right–most conclusion P of the implications $A_0 \supset \ldots \supset A_n \supset P$ and selects this formula only if the right–hand side of the sequent coincides with P, similarly to the search strategy of PROLOG. This algorithm turns out to be faster with sequents which are hard on the algorithm presented in Sect. 4; and vice versa. Thus it might be useful to have several algorithms.

Since our proof is rather simple and elementary, formalising this approach seems promising. Indeed the $\supset$–fragment is formalised in the system MINLOG [10].

2 Notation

We use P, Q to denote atomic formulas and A, $B, \ldots$ to denote formulas. Formulas are generated by atomic formulas, the falsum $\bot$, implications $A \supset B$, conjunctions $A \wedge B$, and disjunctions $A \vee B$. Negation $\neg A$ is treated as an abbreviation for $A \supset \bot$. Lists of formulas are denoted by Γ, Δ and sequents by $\Gamma \Rightarrow A$.

To avoid some parentheses, we write $A \supset B \supset C$ instead of $A \supset (B \supset C)$, $A \wedge B \supset C$ instead of $(A \wedge B) \supset C$, and $A \vee B \supset C$ instead of $(A \vee B) \supset C$.

2.1 Counter–Models

For an introduction to Kripke models, see e.g. Troelstra and van Dalen [11]. A Kripke model $\mathcal{K}$ is a tripel $(K, \leq, \Vdash)$, where K is a non–empty set of worlds denoted by k. We list the properties we need:

- $k \not\Vdash \bot$.
- $k \Vdash A \wedge B$ iff $k \Vdash A$ and $k \Vdash B$.
- $k \Vdash A \vee B$ iff $k \Vdash A$ or $k \Vdash B$.
- $k \Vdash A \supset B$ iff $k' \Vdash A$ implies $k' \Vdash B$ for all $k' \geq k$.

– If $k \Vdash A$ and $k' \geq k$ then $k' \Vdash A$ (monotonicity).

We extend the relation $\Vdash$ in a natural way to lists and write $k \Vdash \Gamma$ if $k \Vdash A$ for each $A \in \Gamma$.

Since $k \Vdash A$ cannot be assumed to be decidable, we have to specialise our semantics. A *Kripke tree* is a non–empty, finite, and finitely branching tree, the nodes of which are labelled by finite sets L of atomic formulas. We use $(L, \vec{\mathcal{K}})$ to denote such a node, where $\vec{\mathcal{K}}$ are its successors. A Kripke tree defines a tripel $(K, \leq, \Vdash)$ by taking K as the set of the nodes, $\leq$ the transitive and reflexive closure of the successor relation, and by $(L, \vec{\mathcal{K}}) \Vdash P$ iff $P \in L$. A Kripke tree is a *Kripke tree model* if the associated tripel $(K, \leq, \Vdash)$ is a Kripke model.

For a Kripke model $\mathcal{K} = (K, \leq, \Vdash)$, we write $\mathcal{K} \Vdash A$ if $k \Vdash A$ holds for each $k \in K$. For a Kripke tree model this is, by monotonicity, equivalent to "the root node forces A". Hence "a node $(L, \vec{\mathcal{K}})$ forces A" coincides with "a tree having the root $(L, \vec{\mathcal{K}})$ forces A". Furthermore, every node of a Kripke tree model is a Kripke tree model itself. Thus we identify a node $(L, \vec{\mathcal{K}})$ with a tree having that root.

A Kripke model $\mathcal{K}$ is a *counter–model* to $\Gamma \Rightarrow A$ if $\mathcal{K} \Vdash \Gamma$ and $\mathcal{K} \nVdash A$. In this case we call the sequent $\Gamma \Rightarrow A$ *refutable*.

We will later use the following lemmata to construct a Kripke tree model and to compute $(L, \vec{\mathcal{K}}) \Vdash A \supset B$ inductively.

Lemma 1. *$(L, \vec{\mathcal{K}})$ is a Kripke tree model iff for each $\mathcal{K} \in \vec{\mathcal{K}}$, $\mathcal{K}$ is Kripke tree model, and $\mathcal{K} \Vdash L$.* □

Lemma 2. *Let $(L, \vec{\mathcal{K}})$ be a Kripke tree model. $(L, \vec{\mathcal{K}}) \Vdash A \supset B$ if and only if $(L, \vec{\mathcal{K}}) \Vdash A$ implies $(L, \vec{\mathcal{K}}) \Vdash B$, and $\mathcal{K} \Vdash A \supset B$ for each $\mathcal{K} \in \vec{\mathcal{K}}$.* □

2.2 Derivations

The proofs in the following sections can easily be carried out by any notation for derivations for the intuitionistic logic. Instead of fixing a notation for derivations, we list the neccessary properties by giving some rules in Fig. 1. These rules have to be read constructively: for each instance of the rule, we can compute a derivation of the conclusion from derivations of all premises. In that case, we say a rule *preserves derivability*.

3 Invertibility

In this section, we will summarise some simple properties of propositional intuitionistic logic so that we can focus on the essence in the next section.

A rule is called *invertible* iff for each instance of the rule, the derivability of the conclusion implies that of all premises. It is well–known that most of the rules are invertible. Although it is not hard to prove this (by induction on derivations), we will not use invertibility. Instead, we will work with a semantic notation. We

$$\frac{}{\Gamma, P, \Delta \Rightarrow P}\,\text{Ax} \qquad \frac{}{\Gamma, \bot, \Delta \Rightarrow A}\,\text{Efq}$$

$$\frac{A, \Gamma \Rightarrow B}{\Gamma \Rightarrow A \supset B}\,\text{R-}\supset \qquad \frac{\Gamma, A \supset B, \Delta \Rightarrow A \quad \Gamma, B, \Delta \Rightarrow C}{\Gamma, A \supset B, \Delta \Rightarrow C}\,\text{L-}\supset$$

$$\frac{\Gamma \Rightarrow A \quad \Gamma \Rightarrow B}{\Gamma \Rightarrow A \wedge B}\,\text{R-}\wedge \qquad \frac{\Gamma, A, B, \Delta \Rightarrow C}{\Gamma, A \wedge B, \Delta \Rightarrow C}\,\text{L-}\wedge$$

$$\frac{\Gamma \Rightarrow A}{\Gamma \Rightarrow A \vee B}\,\text{R-}\vee_l \quad \frac{\Gamma \Rightarrow B}{\Gamma \Rightarrow A \vee B}\,\text{R-}\vee_r \qquad \frac{\Gamma, A, \Delta \Rightarrow C \quad \Gamma, B, \Delta \Rightarrow C}{\Gamma, A \vee B, \Delta \Rightarrow C}\,\text{L-}\vee$$

$$\frac{\Gamma, \Delta \Rightarrow A \quad \Gamma, A, \Delta \Rightarrow B}{\Gamma, \Delta \Rightarrow B}\,\text{cut} \qquad \frac{\Gamma, \Delta \Rightarrow B}{\Gamma, A, \Delta \Rightarrow B}\,\text{weakening}$$

Fig. 1.

say a rule *preserves refutability* if for each instance, given a counter–model to one premise, we can compute a counter–model to the conclusion.

Invertibility and preservation of refutability have a close connection. We suppose soundness for a moment. As soon as we have proved completeness, invertibility will imply preservation of refutability. As soon as we have proved decidability, preservation of refutability will entail invertibility.

Each rule we introduce and which preserves refutability turns out to satisfy the following stronger property, which is very easy to verify.

Definition 3. *A rule* preserves counter–models *if for each instance, a counter–model to at least one premise is also a counter–model to the conclusion.*

Lemma 4. *The rules* R-$\supset$, R-$\wedge$, L-$\wedge$, *and* L-$\vee$ *preserve counter–models.* □

We will eliminate formulas of the form $\bot \supset B$ on the left–hand side of sequents. For this, we introduce the following rule:

$$\frac{\Gamma, \Delta \Rightarrow A}{\Gamma, \bot \supset B, \Delta \Rightarrow A}\,\text{L-}\bot\supset$$

Lemma 5. L-$\bot\supset$ *preserves derivability and counter–models.* □

Furthermore, proofs become slightly shorter if we replace $B_0 \wedge B_1 \supset C$ by $B_0 \supset B_1 \supset C$ and $B_0 \vee B_1 \supset C$ by $B_0 \supset C$ and $B_1 \supset C$. For this, we take the following rules already introduced by Vorob'ev [13].

$$\frac{\Gamma, B_0 \supset B_1 \supset C, \Delta \Rightarrow A}{\Gamma, B_0 \wedge B_1 \supset C, \Delta \Rightarrow A}\,\text{L-}\wedge\supset \qquad \frac{\Gamma, B_0 \supset C, B_1 \supset C, \Delta \Rightarrow A}{\Gamma, B_0 \vee B_1 \supset C, \Delta \Rightarrow A}\,\text{L-}\vee\supset$$

Lemma 6. L-$\wedge\supset$ *and* L-$\vee\supset$ *preserve derivability and counter–models.* □

The following linear degree w will be used in the proof of decidability.

Definition 7. *For formulas we define*

$$\begin{cases} w(P) := w(\bot) := 0 \\ w(A \supset B) := 1 + w(A) + w(B) \\ w(A \wedge B) := 2 + w(A) + w(B) \\ w(A \vee B) := 3 + w(A) + w(B). \end{cases}$$

For lists of formulas we define $w(B_1, \ldots, B_n) := w(B_1) + \ldots + w(B_n)$ *and for sequents* $w(\Gamma \Rightarrow A) := w(\Gamma) + w(A)$.

Each instance of any premise of the rules R-$\supset$, R-$\wedge$, R-$\vee$, L-$\vee$, L-$\bot\supset$, and L-$\wedge\supset$ has a smaller w–degree than the corresponding instance of the conclusion. This does not hold for L-$\vee\supset$ if the instance of C is a composed formula. To avoid this, we use the following rule:

$$\frac{\Gamma, B \supset P, P \supset C, \Delta \Rightarrow A}{\Gamma, B \supset C, \Delta \Rightarrow A}\ \text{L-S}\ ,\ \text{provided } P \text{ does not occur in the conclusion.}$$

Lemma 8. L-S *preserves derivability and counter–models.*

Proof. For derivability see Hudelmaier [7]. Obviously, L-S preserves counter–models. □

Now combining L-$\vee\supset$ and L-S

$$\dfrac{\dfrac{\Gamma, B_0 \supset P, B_1 \supset P, P \supset C, \Delta \Rightarrow A}{\Gamma, B_0 \vee B_1 \supset P, P \supset C, \Delta \Rightarrow A}\ \text{L-S}}{\Gamma, B_0 \vee B_1 \supset C, \Delta \Rightarrow A}\ \text{L-}\vee\supset$$

we verify that each instance of the upper sequent has a smaller w–degree than the lower sequent.

At first sight, the rule L-S appears to increase the non–determinism by replacing one formula $B \supset C$ by two formulas $B \supset P$ and $P \supset C$, but, as it turns out, the second formula will only be considered if B is derived.

In general, the rules L-$\supset$ and R-$\vee$ are neither invertible nor do they preserve refutability or counter–models. L-$\supset$ is known to be semi–invertible, i.e. for each instance, the derivability of the conclusion $\Gamma, A \supset B, \Delta \Rightarrow C$ implies that of the right premise $\Gamma, B, \Delta \Rightarrow C$. We will not use this property; instead, we will use the following property.

Lemma 9. *If* $\mathcal{K}$ *is a counter–model to* $\Gamma, B, \Delta \Rightarrow C$*, then also to* $\Gamma, A \supset B, \Delta \Rightarrow C$. □

In the case that the premise of an implication is atomic and is in the left–hand side already, we get the following rule(cf. Hudelmaier [6, 7] and Dyckhoff [1]).

$$\frac{\Gamma, C, \Delta \Rightarrow A}{\Gamma, P \supset C, \Delta \Rightarrow A}\ \text{L-}P\supset,\ \text{provided } P \in \Gamma, \Delta$$

Corollary 10. L-$P\supset$ *preserves derivability and counter–models.*

Proof. By L-$\supset$ and Ax, we get that L-$P\supset$ preserves derivability. Lemma 9 says that L-$P\supset$ preserves counter–models. □

The above section can be summed up in the next lemma applying the following definition, which is an extension of Dyckhoff's [1].

Definition 11. *A sequent $\Gamma \Rightarrow A$ is called* irreducible *if Γ contains only atomic formulas*[1]*, or formulas of the form $P \supset B$ where P is not in Γ, or else formulas of the form $(B_0 \supset B_1) \supset C$. Furthermore, we require that A is either a disjunction, or falsum, or else an atomic formula not in Γ. A sequent is called* reducible *if it is not irreducible.*

Thus a sequent $\Gamma \Rightarrow A$ is irreducible iff $\Gamma \Rightarrow A$ is neither an instance of a conclusion of one of the counter–models preserving rules introduced so far(L-S only if B is a disjunction and C is a composed formula), nor an instance of a conclusion of Ax and Efq. We will sometimes denote a sequent by $\mathcal{S}$.

Lemma 12. *For each reducible $\Gamma \Rightarrow A$ one can find $\mathcal{S}_1, \ldots, \mathcal{S}_n$ such that*
(a) $w(\mathcal{S}_i) < w(\Gamma \Rightarrow A)$ for each i.
(b) If each $\mathcal{S}_i$ is derivable, then so is $\Gamma \Rightarrow A$.
(c) If $\mathcal{K}$ is a counter–model to at least one $\mathcal{S}_i$ then also to $\Gamma \Rightarrow A$.

Proof. Case $\Gamma = \Gamma_0, B_0 \wedge B_1, \Gamma_2$. Let $\mathcal{S}_1, \mathcal{S}_2$ be the left respectively right premise of L-$\wedge$ where $B_0 \wedge B_1$ is the principal formula. (a) is obvious. (b) holds since L-$\wedge$ preserves derivability, and (c) holds because L-$\wedge$ preserves counter–models.

Case $\Gamma = \Gamma_0, (B_0 \vee B_1) \supset C, \Gamma_2$. If C is a composed formula, then we proceed by combining L-$\vee\supset$ and L-S as described above. If C is atomic or $\bot$, we obtain our statement by L-$\vee\supset$.

The *remaining cases* follow similarly using the corresponding rules, all of which preserve counter–models. □

By a similar proof, we obtain a variant of the previous lemma, which will be motivated in Theorem 16.

Lemma 13. *Let A be a disjunction and B an arbitrary formula. For each reducible $\Gamma_1, \Gamma_2 \Rightarrow A$ one can find $\mathcal{S}_1, \ldots, \mathcal{S}_n$ such that*
(a) The left–hand side of each $\mathcal{S}_i$ has a smaller w–degree than $\Gamma_1, A \supset B, \Gamma_2$.
(b) If each $\mathcal{S}_i$ is derivable, then so is $\Gamma_1, A \supset B, \Gamma_2 \Rightarrow A$.
(c) If $\mathcal{K}$ is a counter–model to at least one $\mathcal{S}_i$ then also to $\Gamma_1, A \supset B, \Gamma_2 \Rightarrow A$.
(d) Each $\mathcal{S}_i$ is of the form $\Gamma_{1,i}, A \supset B, \Gamma_{2,i} \Rightarrow A$. □

[1] Note that $\bot$ is by definition not atomic.

4 Forward chaining

How can we deal with the implication in the context $\Gamma, A \supset B, \Delta \Rightarrow C$? One tactic is to search for a derivation of $\Gamma, A \supset B, \Delta \Rightarrow A$ and then to go on with $\Gamma, B, \Delta \Rightarrow C$. This can be called *forward chaining*. The problem is how to enforce termination without destroying completeness.

The first premise $\Gamma, A \supset B, \Delta \Rightarrow A$ has a particular form: the right–hand side A occurs as a premise of an implication on the left–hand side. If A is a composed formula, then, as observed by Hudelmaier, the problem can be reduced to a smaller one of the same form. For this, he introduced the rules $GI2{\rightarrow}$, $GI2{\wedge}$, and $GI2{\vee}$ in [7]. We rename $GI2{\rightarrow}$ to L2-$\supset\supset$ and $GI2{\wedge}$ to L2-$\wedge\supset$. In $GI2{\vee}$, the rule L-S is incorporated. We separate L-S and call the remaining part L2-$\vee\supset$.

$$\frac{A_0, \Gamma, A_1 \supset B, \Delta \Rightarrow A_1}{\Gamma, (A_0 \supset A_1) \supset B, \Delta \Rightarrow A_0 \supset A_1}\ \text{L2-}\supset\supset$$

$$\frac{\Gamma, A_0 \supset B, \Delta \Rightarrow A_0 \quad \Gamma, A_1 \supset B, \Delta \Rightarrow A_1}{\Gamma, A_0 \wedge A_1 \supset B, \Delta \Rightarrow A_0 \wedge A_1}\ \text{L2-}\wedge\supset$$

$$\frac{\Gamma, A_0 \supset B, A_1 \supset B, \Delta \Rightarrow A_i}{\Gamma, A_0 \vee A_1 \supset B, \Delta \Rightarrow A_0 \vee A_1}\ \text{L2-}\vee\supset_i, \text{ where } i \in \{0, 1\}$$

Lemma 14. L2-$\supset\supset$ *and* L2-$\wedge\supset$ *preserves derivability and counter–models,* L2-$\vee\supset$ *preserves derivability (only).*

Proof. To see that L2-$\supset\supset$ preserves derivability, use cut and a derivation of $A_0, (A_0 \supset A_1) \supset B \Rightarrow A_1 \supset B$. To see that L2-$\wedge\supset$ preserves derivability, use cut and a derivation of $(A_0 \supset B) \supset A_0, (A_1 \supset B) \supset A_1, A_0 \wedge A_1 \supset B \Rightarrow A_0 \wedge A_1$. To prove that L2-$\vee\supset$ preserves derivability, combine R-$\vee$ and L-$\vee\supset$.

The proof that L2-$\supset\supset$ and L2-$\wedge\supset$ preserve counter–models is easy and is therefore left to the reader. □

Note that L2-$\vee\supset$ is neither invertible nor refutability–preserving.

How can we deal with the implications $P \supset A$ where P is atomic? If P is in the left–hand side, we apply L-$P\supset$. What do we have to do if P is not in the left–hand side? Then we need not consider these formulas, as the following lemma shows. This was already observed by Vorob'ev [13], but we give a much simpler proof.

Lemma 15. *Let $\Gamma \Rightarrow A$ be an irreducible sequent. Let $\vec{\mathcal{K}}$ be Kripke tree models so that $\mathcal{K}' \Vdash \Gamma$ for each $\mathcal{K}' \in \vec{\mathcal{K}}$, and for each formula $(B_0 \supset B_1) \supset C$ in Γ there is a counter–model in $\vec{\mathcal{K}}$ to $\Gamma \Rightarrow B_0 \supset B_1$. Furthermore, if A is a disjunction $A_0 \vee A_1$, then suppose that there are counter–models $\mathcal{K}_0$, $\mathcal{K}_1$ in $\vec{\mathcal{K}}$ to $\Gamma \Rightarrow A_0$, $\Gamma \Rightarrow A_1$ respectively. Then $\mathcal{K} := (\{P : P \in \Gamma\}, \vec{\mathcal{K}})$ is a counter–model to $\Gamma \Rightarrow A$.*

Proof. As $\mathcal{K}' \Vdash \Gamma$ implies $\mathcal{K}' \Vdash \{P : P \in \Gamma\}$, Lemma 1 provides that $\mathcal{K}$ is a Kripke tree model.

Let $D \in \Gamma$. We have to show $\mathcal{K} \Vdash D$. Since $\Gamma \Rightarrow A$ is irreducible, D is either an atom, or an implication $B \supset C$ where B is either $B_0 \supset B_1$, or else an atom not in Γ. If D is an atomic formula, then $D \in \{P : P \in \Gamma\}$ and hence $\mathcal{K} \Vdash D$. If $D = B \supset C$, then by Lemma 2 and because of $\mathcal{K}' \Vdash \Gamma$, it is sufficient to show that $\mathcal{K} \Vdash B$ implies $\mathcal{K} \Vdash C$. We show $\mathcal{K} \nVdash B$. If B is an atom not in Γ, then $B \notin \{P : P \in \Gamma\}$ and hence $\mathcal{K} \nVdash B$. Otherwise $B = B_0 \supset B_1$ and there is a $\mathcal{K}'$ in $\vec{\mathcal{K}}$ such that $\mathcal{K}' \nVdash B$, implying $\mathcal{K} \nVdash B$ by monotonicity.

It remains to show $\mathcal{K} \nVdash A$. Since $\Gamma \Rightarrow A$ is irreducible, A is either an atom not in Γ, or $\bot$, or an disjunction $A_0 \vee A_1$. In the first case we have $A \notin \{P : P \in \Gamma\}$ and hence $\mathcal{K} \nVdash A$. If $A = \bot$, then $\mathcal{K} \nVdash A$ by definition. Otherwise $A = A_0 \vee A_1$. Here there are $\mathcal{K}_0, \mathcal{K}_1 \in \vec{\mathcal{K}}$ where $\mathcal{K}_i \nVdash A_i$. By monotonicity we get $\mathcal{K} \nVdash A_0$ and $\mathcal{K} \nVdash A_1$. Hence $\mathcal{K} \nVdash A_0 \vee A_1$ by definition. □

Particularly, if $\Gamma \Rightarrow P/\bot$ is irreducible and Γ does not contain formulas of the form $(B_0 \supset B_1) \supset C$, then $(\{P : P \in \Gamma\}, \varepsilon)$ is a counter–model to $\Gamma \Rightarrow P/\bot$.

The previous lemma is the computational content of the refutation rules

$$\frac{\Gamma \not\Rightarrow B_1 \supset C_1 \quad \cdots \quad \Gamma \not\Rightarrow B_n \supset C_n}{\Gamma \not\Rightarrow P/\bot}\, n \geq 0$$

$$\frac{\Gamma \not\Rightarrow B_1 \supset C_1 \quad \cdots \quad \Gamma \not\Rightarrow B_n \supset C_n \quad \Gamma \not\Rightarrow A_0 \quad \Gamma \not\Rightarrow A_1}{\Gamma \not\Rightarrow A_0 \vee A_1}\, n \geq 0$$

provided that $\Gamma \Rightarrow P$, $\Gamma \Rightarrow \bot$, and $\Gamma \Rightarrow A_0 \vee A_1$ respectively, are irreducible, and where $(B_1 \supset C_1) \supset D_1, \ldots, (B_n \supset C_n) \supset D_n$ are all nested implications in Γ. Separating L2-$\supset\supset$ from the rule (11) given by Pinto and Dyckhoff [9], our rules coincide with a non–multi–succedent version of theirs. While Pinto and Dyckhoff work with the rules themselves, we work with the computational content of the rules. Thus we do not have to construct the counter–models in a second step, unlike the approach of Pinto and Dyckhoff.

Now we are ready to prove the decidability of the intuitionistic propositional logic. This will be done in part (i) of the following theorem. For the proof, we also need part (ii). Note that in (ii) the right–hand side of the sequent does not contribute to the w–degree.

Theorem 16. *Let n be any natural number.*
(i) Each $\Gamma \Rightarrow A$ with $w(\Gamma, A) \leq n$ has either a derivation or a counter–model.
(ii) Each $\Gamma_1, A \supset B, \Gamma_2 \Rightarrow A$ with $w(\Gamma_1, A \supset B, \Gamma_2) \leq n$ has either a derivation or a counter–model.

Proof. We proceed by simultaneous, progressive induction on n, i.e. let n be given and we are allowed to use the induction hypothesis for (i) and (ii) for all $m < n$. First we prove (i), then we prove (ii). We write IH for "induction hypothesis" .

As for (i), we proceed by case analysis on whether (1) the sequent is reducible, or (2) the sequent is an instance of Efq or Ax, or (3) the IH(ii) on the left premise of an instance of a combination of L2-$\supset\supset$ and L-$\supset$ yields a derivation for at least one formula $(B_0 \supset B_1) \supset C$ of Γ, or (4) A is a disjunction and the IH(i) on at least one premise of R-$\vee$ provides a derivation, or else the remaining case.

Case 1. $\Gamma \Rightarrow A$ is reducible. Let $\mathcal{S}_1, \ldots, \mathcal{S}_n$ be the sequents obtained from Lemma 12. The IH(i) on each $\mathcal{S}_i$ yields either a derivation for all $\mathcal{S}_i$ or a counter–model to at least one $\mathcal{S}_i$. In the first case, we get a derivation of $\Gamma \Rightarrow A$ by (b) of Lemma 12. In the second case, we have already got a counter–model to $\Gamma \Rightarrow A$ according to (c) of Lemma 12.

Case 2. $\bot \in \Gamma$ or A is atomic and in Γ. Here we get a derivation by Efq or Ax respectively.

Case 3. For a partition $\Gamma = \Gamma_1, (B_0 \supset B_1) \supset C, \Gamma_2$ the IH(ii) on $B_0, \Gamma_1, B_1 \supset C, \Gamma_2 \Rightarrow B_1$ yields a derivation. Here we apply L2-$\supset\supset$ to obtain a derivation of $\Gamma_1, (B_0 \supset B_1) \supset C, \Gamma_2 \Rightarrow B_0 \supset B_1$. Moreover, the IH(i) on $\Gamma_1, C, \Gamma_2 \Rightarrow A$ yields a derivation or a counter–model. In the first case, L-$\supset$ provides a derivation of $\Gamma \Rightarrow A$. In the second case, we have already got a counter–model to that sequent by Lemma 9.

Case 4. $A = A_0 \vee A_1$ and the IH(i) on $\Gamma \Rightarrow A_0$ or on $\Gamma \Rightarrow A_1$ yields a derivation. Here, R-$\vee$ provides a derivation of $\Gamma \Rightarrow A$.

Remaining case. For every partition $\Gamma = \Gamma_1, (B_0 \supset B_1) \supset C, \Gamma_2$ the IH(ii) yields a counter–model to $B_0, \Gamma_1, B_1 \supset C, \Gamma_2 \Rightarrow B_1$ and if $A = A_0 \vee A_1$, the IH(i) on both $\Gamma \Rightarrow A_0$ and $\Gamma \Rightarrow A_1$ also yields two counter–models. Let $\vec{\mathcal{K}}$ be all these counter–models. Since L2-$\supset\supset$ preserves counter–models, counter–models to $B_0, \Gamma_1, B_1 \supset C, \Gamma_2 \Rightarrow B_1$ are also counter–models to $\Gamma \Rightarrow B_0 \supset B_1$. Hence, by Lemma 15, $(\{P : P \in \Gamma\}, \vec{\mathcal{K}})$ is a counter–model to $\Gamma \Rightarrow A$. This completes the proof of (i).

As for (ii) we proceed by case analysis on the form of A.

Case A atomic or $\bot$. As we have already proved (i) for n, we can apply (i) to the sequent $\Gamma_1, A \supset B, \Gamma_2 \Rightarrow A$.

Case $A = A_0 \supset A_1$. The IH(ii) on $A_0, \Gamma_1, A_1 \supset B, \Gamma_2 \Rightarrow A_1$ yields a derivation or a counter–model. Since L2-$\supset\supset$ preserves derivability and counter–models, we obtain a derivation of $\Gamma_1, A_0 \wedge A_1 \supset B, \Gamma_2 \Rightarrow A_0 \wedge A_1$ or we have already got a counter–model to that sequent.

Case $A = A_0 \wedge A_1$. Similarly using L2-$\wedge\supset$.

Case $A = A_0 \vee A_1$. Here we have to consider similar cases as in (i).

Subcase 1. $\Gamma_1, \Gamma_2 \Rightarrow A$ is not irreducible. Let $\mathcal{S}_1, \ldots, \mathcal{S}_n$ be the sequents obtained from Lemma 13. The IH(ii) on each $\mathcal{S}_i$ yields either a derivation for all $\mathcal{S}_i$ or a counter–model to at least one $\mathcal{S}_i$. We proceed as in (i) case 1.

Subcase 2. $\bot \in \Gamma$ or A is atomic and in Γ. Analogously to (i) case 2.

Subcase 3. For a formula $(C_0 \supset C_1) \supset D$ of Γ_1, Γ_2, the IH(ii) yields a derivation as in (i) case 3. We proceeds as in (i) case 3, replacing "IH(i)" by "IH(ii)".

Subcase 4a. B is atomic or $\bot$ and the IH(ii) on $\Gamma_1, A_0 \supset B, A_1 \supset B, \Gamma_2 \Rightarrow A_0$ or on $\Gamma_1, A_0 \supset B, A_1 \supset B, \Gamma_2 \Rightarrow A_1$ yields a derivation. Then L2-$\vee\supset$ provides a derivation of the sequent under consideration.

Subcase 4b. B is a composed formula and the IH(ii) on $\Gamma_1, A_0 \supset P, A_1 \supset P, P \supset B, \Gamma_2 \Rightarrow A_0$ or on $\Gamma_1, A_0 \supset P, A_1 \supset P, P \supset B, \Gamma_2 \Rightarrow A_1$ where P is a new atomic formula yields a derivation. Using L2-$\vee\supset$ and L-S provides a derivation.

In the *remaining subcase* we construct a counter–model as in (i), again replacing "IH(i)" by "IH(ii)". □

This proof describes an algorithm returning a derivation or a counter–model for a sequent $\Gamma \Rightarrow A$. The algorithm consists of two parts, say search(i) and search(ii). After applying bottom–up all rules preserving counter–models, if the new sequent is not an instance of the conclusion of the rules Ax or Efq, search(i) picks up all the formulas $(B_0 \supset B_1) \supset C$ one after the other and applies search(ii) to $B_0, \Gamma_1, B_1 \supset C, \Gamma_2 \Rightarrow B_1$ where $\Gamma = \Gamma_1, (B_0 \supset B_1) \supset C, \Gamma_2$. If a derivation is returned for one of these sequents, search(i) is called on $\Gamma_1, C, \Gamma_2 \Rightarrow A$. Otherwise, search(i) tries to apply R-$\vee$; if this does not succeed either, a counter–model is returned.

Search(ii) applies L2-$\supset\supset$ and L2-$\wedge\supset$ until A is an atom or a disjunction $A_0 \vee A_1$. In the first case, search(i) is applied. In the second case, all nested implications are picked up as in search(i), and if this fails, L2-$\vee\supset$ is tested (if B is a composed formula, only in combination with L-S). If this does not succeed either, a counter–model is returned.

Immediately, we see that the number of recursions is bounded by twice the w–degree of the input sequent. If we do not count the call of search(i) in search(ii) on the identical sequent, the number of recursions is even bounded by the w–degree of the input. Hence we obtain the following estimate.

Corollary 17. *If a sequent $\Gamma \Rightarrow A$ is refutable, then the algorithm described above returns a counter–model with height less than $w(\Gamma \Rightarrow A) + 1$.* □

While Hudelmaier [7], and Pinto and Dyckhoff [9] essentially proved completeness and termination for their calculi LF, LJT* respectively, allowing one to extract a decision algorithm for intuitionistic propositional logic, the present work starts off with a proof of decidability from which one can read off a decision procedure. However, when dropping the task of producing counter–models, this decision procedure coincides with that for LF. Furthermore, if we proved decidability using the computational contents of the rules given by Pinto and Dyckhoff, we would obtain an algorithm similar to that for LJT*, when dropping counter–models, and similar to CRIP, when dropping derivations.

5 Pruning the search tree

In contrast to Underwood's [12] approach, where the counter–model is constructed after the whole search fails, the present approach constructs local counter–models during the search. These local counter–models can be used to reduce the non–determinism. For example we consider (cf. Fig. 2) an irreducible sequent $(B_0 \supset B_1) \supset B_2, \Gamma, (D_0 \supset D_1) \supset D_2 \Rightarrow A$. To derive that sequent, we select $(B_0 \supset B_1) \supset B_2$ and search for a derivation of $(B_0 \supset B_1) \supset B_2, \Gamma, (D_0 \supset D_1) \supset D_2 \Rightarrow$

$B_0 \supset B_1$. Now L2-$\supset\supset$ reduces this to the search for $B_0, B_1 \supset B_2, \Gamma, (D_0 \supset D_1) \supset D_2 \Rightarrow B_1$. For simplicity, we assume that B_1 is atomic and that the sequent is irreducible. Furthermore, we assume that selecting each formula $(C_0 \supset C_1) \supset C_2$ of Γ yields a counter–model

$$\mathcal{K}_C \text{ to } B_0, B_1 \supset B_2, \Gamma, (D_0 \supset D_1) \supset D_2 \Rightarrow C_0 \supset C_1.$$

Finally we select $(D_0 \supset D_1) \supset D_2$ and apply L2-$\supset\supset$ bottom–up.

$$\dfrac{\dfrac{\dfrac{D_0, B_0, B_1 \supset B_2, \Gamma, D_1 \supset D_2 \Rightarrow D_1}{\cdots} \quad B_0, B_1 \supset B_2, \Gamma, D_2 \Rightarrow B_1}{B_0, B_1 \supset B_2, \Gamma, (D_0 \supset D_1) \supset D_2 \Rightarrow B_1} \quad \dfrac{}{\cdots} \quad B_2, \Gamma, D^{\dagger} \Rightarrow A}{(B_0 \supset B_1) \supset B_2, \Gamma, (D_0 \supset D_1) \supset D_2 \Rightarrow A}$$

$^{\dagger}\ D = (D_0 \supset D_1) \supset D_2$

Fig. 2.

We have to search for the sequents

(1a) $D_0, B_0, B_1 \supset B_2, \Gamma, D_1 \supset D_2 \Rightarrow D_1$,
(2a) $B_0, B_1 \supset B_2, \Gamma, D_2 \Rightarrow B_1$, and
(3a) $B_2, \Gamma, (D_0 \supset D_1) \supset D_2 \Rightarrow A$.

We assume that these sequents are irreducible. For the search we have to select each formula $(C_0 \supset C_1) \supset C_2$ of Γ again, unless we have a counter–model to

(1b) $B_0, B_1 \supset B_2, \Gamma, D_0, D_1 \supset D_2 \Rightarrow C_0 \supset C_1$,
(2b) $B_0, B_1 \supset B_2, \Gamma, D_2 \Rightarrow C_0 \supset C_1$,
(3b) $B_2, \Gamma, (D_0 \supset D_1) \supset D_2 \Rightarrow C_0 \supset C_1$, respectively.

Possibly $\mathcal{K}_C$ is a counter–model to one of these sequents; we only have to check if

(1c) $\mathcal{K}_C \Vdash D_0$,[2]
(2c) $\mathcal{K}_C \Vdash D_2$,
(3c) $\mathcal{K}_C \Vdash B_2$, respectively.

If we do not obtain a derivation of (1a) or (2a), we have to select each formula $(C_0 \supset C_1) \supset C_2$ of $\Gamma, (D_0 \supset D_1) \supset D_2$ in order to search for either a derivation of or a counter–model to $(B_0 \supset B_1) \supset B_2, \Gamma, (D_0 \supset D_1) \supset D_2 \Rightarrow C_0 \supset C_1$. Yet, $\mathcal{K}_C$ are counter–models to those sequents already!

In other steps of the algorithm, we can proceed similarly. In this way, we can prune the search tree. To do this, we have to pass the counter–models up, left, and down, which cannot be done by a sequent calculus.

[2] Since $\mathcal{K}_C \Vdash (D_0 \supset D_1) \supset D_2$ implies $\mathcal{K}_C \Vdash D_1 \supset D_2$.

Of course, in the worst case, we have to consider the whole tree in order to verify that a Kripke tree model forces a formula. Thus pruning the search tree in this way does not always reduce the runtime for every kind of pruning. But if the formula contains no $\supset$, we only have to consider the root of a Kripke tree.

Moreover, although an automatic theorem prover can handle most sequents very fast, an automatic theorem prover may exceed an acceptable runtime, since deciding intuitionistic propositional logic is PSPACE–hard. In this case an interactive prover may help the user by indicating which formulas he or she need not select. Here the additional runtime of checking whether a Kripke tree model forces a formula is almost always acceptable.

6 Backward Chaining

We will now give an alternative proof of decidability. The proof is much simpler if we restrict ourselves to the $\supset$–fragment. We will do so now. In this section, formulas are only generated by atomic formulas and implications. By abuse of notation, we use $\vec{A} \supset B$ to denote $A_1 \supset \ldots \supset A_n \supset B$, where the list $\vec{A}$ is possibly empty; in this case, we identify $\vec{A} \supset B$ with B. In the $\supset$–fragment, every formula B has the form $(\vec{A}_1 \supset Q_1) \supset \ldots \supset (\vec{A}_k \supset Q_k) \supset Q$ for $k \geq 0$. Q is called head of B. Since R-$\supset$ preseres derivability and counter–models, we only have to consider sequents where the right–hand side is atomic.

To derive a sequent $\Gamma \Rightarrow P$, we want to select a formula B from the context Γ only if the head of B is equal to P. We call this *backward chaining.*

We will use a generalisation of L2-$\wedge\supset$ and L2-$\supset\supset$:

$$\frac{\vec{A}_1, \Gamma_1, Q_1 \supset Q, \Gamma_2 \Rightarrow Q_1 \qquad \ldots \qquad \vec{A}_k, \Gamma_1, Q_k \supset Q, \Gamma_2 \Rightarrow Q_k}{\Gamma_1, (\vec{A}_1 \supset Q_1) \supset \ldots \supset (\vec{A}_k \supset Q_k) \supset Q, \Gamma_2 \Rightarrow Q} \text{ L3-}\supset^*$$

Note that in the case $k > 1$ or $k = 1$ and $\vec{A}_1 \neq \varepsilon$, each premise has a lower w–degree than the conclusion. Also note that, in the $\supset$–fragment, w coincides with the total number of $\supset$'s.

Lemma 18. L3-$\supset^*$ *preserves derivability and counter–models.*

Proof. To prove that L3-$\supset^*$ preserves derivability, we assume derivations of $\vec{A}_i, \Gamma_1, Q_i \supset Q, \Gamma_2 \Rightarrow Q_i$ for each i. By induction on the length of $\vec{A}_i$ and by L2-$\supset\supset$, we obtain derivations of $\Gamma_1, (\vec{A}_i \supset Q_i) \supset Q, \Gamma_2 \Rightarrow \vec{A}_i \supset Q_i$. By induction on k and by L2-$\wedge\supset$, we get a derivation of $\Gamma_1, (\vec{A}_1 \supset Q_1) \wedge \ldots \wedge (\vec{A}_k \supset Q_k) \supset Q, \Gamma_2 \Rightarrow Q$. Since $(\vec{A}_1 \supset Q_1) \supset \ldots \supset (\vec{A}_k \supset Q_k) \supset Q \Rightarrow (\vec{A}_1 \supset Q_1) \wedge \ldots \wedge (\vec{A}_k \supset Q_k) \supset Q$ is derivable, cut yields a derivation of $\Gamma_1, (\vec{A}_1 \supset Q_1) \supset \ldots \supset (\vec{A}_k \supset Q_k) \supset Q, \Gamma_2 \Rightarrow Q$.

To prove that L-$\supset^*$ preserves counter–models, one either proceeds in a similar way, or one proves it directly. □

Definition 19. *A formula B is a* properly nested implication *if B is of the form $(\vec{A}_1 \supset Q_1) \supset \ldots \supset (\vec{A}_k \supset Q_k) \supset Q$ where $k > 1$, or $k = 1$ and $\vec{A}_1 \neq \varepsilon$.*

If B is not a properly nested implication, i.e. $B = Q_1 \supset Q$, we do not reach our goal completely. But we can deal with these formulas by means of a simple loop–check.

Definition 20. *A* chain in Γ from P_1 to P_n *is a list of implications* $P_1 \supset P_2, P_2 \supset P_3, \ldots, P_{n-1} \supset P_n$ *where each formula* $P_i \supset P_{i+1}$ *is in* Γ*. The empty list is a chain in* Γ *from* P *to* P*, for every* P*.*

Lemma 21. *Suppose there is a chain in* Γ *from* P_1 *to* P_n*. If* $\Gamma \Rightarrow P_1$ *is derivable, then so is* $\Gamma \Rightarrow P_n$*.* □

Definition 22. *An atomic formula* P_1 *is called* significant *for* $\Gamma \Rightarrow P$ *if there is a chain from* P_1 *to* P *in* Γ*.*

Lemma 23. *Let* $P_1 \supset P_2 \in \Gamma$*. If* P_2 *is significant for* $\Gamma \Rightarrow P_n$*, then so is* P_1*.* □

Now we give an alternative construction of a counter–model.

Lemma 24. *Let a sequent* $\Gamma \Rightarrow P$ *and a list of Kripke tree models* $\vec{\mathcal{K}}$ *be given. Let* L *be the set of all atomic formulas significant for* $\Gamma \Rightarrow P$*. Suppose* $L \cap \Gamma = \emptyset$ *and* $\mathcal{K}' \Vdash \Gamma$ *for each* $\mathcal{K}'$ *in* $\vec{\mathcal{K}}$*. Furthermore, suppose that for each properly nested implication* $(\vec{A}_1 \supset Q_1) \supset \ldots \supset (\vec{A}_k \supset Q_k) \supset Q$ *in* Γ *where* Q *is significant for* $\Gamma \Rightarrow P$*, there is a* $\mathcal{K}' \in \vec{\mathcal{K}}$ *such that* $\mathcal{K}'$ *is a counter–model to* $\Gamma \Rightarrow Q$*. Then* $\mathcal{K} := (\bigcap \pi_{\text{left}}(\vec{\mathcal{K}}) - L, \vec{\mathcal{K}})$ *is a counter–model to* $\Gamma \Rightarrow P$*, where* $\bigcap \pi_{\text{left}}(\vec{\mathcal{K}})$ *denotes the intersection of the labels of the roots of* $\vec{\mathcal{K}}$*.*[3]

Proof. First note that $\mathcal{K}$ is a Kripke tree model by Lemma 1. Furthermore, we have $\mathcal{K} \nVdash P$, since $P \in L$. It remains to show $\mathcal{K} \Vdash \Gamma$. Assume $B \in \Gamma$.

Case B is atomic. By assumption $B \notin L$ and $\mathcal{K}' \Vdash B$, i.e. by definition $B \in \pi_{\text{left}}(\mathcal{K}')$ for each $\mathcal{K}'$. Hence $B \in \bigcap \pi_{\text{left}}(\vec{\mathcal{K}}) - L$.

Case $B = P_1 \supset P_2$. Due to $\mathcal{K}' \Vdash \Gamma$ for each $\mathcal{K}'$ and Lemma 2, we only have to show that $\mathcal{K} \Vdash P_1$ implies $\mathcal{K} \Vdash P_2$. Assume $\mathcal{K} \Vdash P_1$. Then by definition $P_1 \in \bigcap \pi_{\text{left}}(\vec{\mathcal{K}})$ and $P_1 \notin L$. The former is equivalent to $\mathcal{K}' \Vdash P_1$ for each $\mathcal{K}'$. By assumption $\mathcal{K}' \Vdash \Gamma$, particularly $\mathcal{K}' \Vdash P_1 \supset P_2$, it follows that $\mathcal{K}' \Vdash P_2$. Hence $P_2 \in \bigcap \pi_{\text{left}}(\vec{\mathcal{K}})$. The latter is equivalent to P_1 non–significant. By Lemma 23, P_2 is non–significant as well, i.e. $P_2 \notin L$.

Case $B = (\vec{A}_1 \supset Q_1) \supset \ldots \supset (\vec{A}_k \supset Q_k) \supset Q$ is a properly nested implication: By Lemma 2 we only have to show that $\mathcal{K} \Vdash \vec{A}_i \supset Q_i$ for each i implies $\mathcal{K} \Vdash Q$.

Subcase Q is significant. Then there is a $\mathcal{K}'$ so that $\mathcal{K}'$ is a counter–model to $\Gamma \Rightarrow Q$. Since $\mathcal{K}' \Vdash B$ and $\mathcal{K}' \Vdash \vec{A}_i \supset Q_i$ for each i using monotonicity, we get $\mathcal{K}' \Vdash Q$. This yields absurdity by $\mathcal{K}' \nVdash Q$ and hence $\mathcal{K} \nVdash \vec{A}_i \supset Q_i$.

Subcase Q is not significant, i.e. $Q \notin L$. Assume that $\mathcal{K} \Vdash \vec{A}_i \supset Q_i$ for each i. By monotonicity $\mathcal{K}' \Vdash \vec{A}_i \supset Q_i$ for each $\mathcal{K}'$ and for each i. By the assumption $\mathcal{K}' \Vdash \Gamma$, particularly $\mathcal{K}' \Vdash B$, it follows $\mathcal{K}' \Vdash Q$. Hence $Q \in \bigcap \pi_{\text{left}}(\vec{\mathcal{K}}) - L$, i.e. $\mathcal{K} \Vdash Q$ by definition. □

[3] To avoid running into infinite sets, we interpret $\bigcap \pi_{\text{left}}(\varepsilon)$ as the set of all atomic formulas occurring as sub–formulas in $\Gamma \Rightarrow A$. Here a more elegant approach would be to work with $(L, \vec{\mathcal{K}}) \Vdash P$ iff $P \notin L$ instead.

In the previous section, the labelling of the nodes was minimal in the sense that each label has to contain at least all atomic formulas in Γ.

By monotonicity the label must be included in $\bigcap \pi_{\mathrm{left}}(\vec{\mathcal{K}})$. Furthermore, the label must not contain any significant atomic formula. Hence, now the labelling is maximal. The advantage of this labelling is that the trees are smaller, since there are fewer possibilities to enlarge the label. To see this, note that a Kripke tree model with each node having identical labels is equivalent to the leaf with the same label; hence enlarging the label is essential.

Now we give the alternative proof of decidability.

Theorem 25. *Each sequent $\Gamma \Rightarrow P$ has a derivation or a counter–model.*

Proof. We proceed by progressive induction on $w(\Gamma)$.

Case. There is a significant Q in Γ. Then there is a chain from Q to P in Γ by definition. Lemma 21 provides a suitable derivation.

Case. For a partition $\Gamma = \Gamma_1, B, \Gamma_2$ where $B = (\vec{A}_1 \supset Q_1) \supset \ldots \supset (\vec{A}_k \supset Q_k) \supset Q$ is a properly nested implication and where Q is significant, the induction hypothesis on $\vec{A}_i, \Gamma_1, Q_i \supset Q, \Gamma_2 \Rightarrow Q_i$ yields a derivation for each $1 \leq i \leq k$. Since L3-$\supset^*$ preserves derivability, we obtain a derivation of $\Gamma \Rightarrow Q$. Lemma 21 yields a derivation of $\Gamma \Rightarrow P$ as required.

Remaining case. Γ has no significant atom and for each partition $\Gamma = \Gamma_1, B, \Gamma_2$ where $B = (\vec{A}_1 \supset Q_1) \supset \ldots \supset (\vec{A}_k \supset Q_k) \supset Q$ is a properly nested formula and where Q is significant, the induction hypothesis yields a counter–model to $\vec{A}_i, \Gamma_1, Q_i \supset Q, \Gamma_2 \Rightarrow Q_i$ for an i. Since L3-$\supset^*$ preserves counter–models, this is also a counter–model to $\Gamma \Rightarrow Q$. Let $\vec{\mathcal{K}}$ be all these counter–models. By Lemma 24, $(\bigcap \pi_{\mathrm{left}}(\vec{\mathcal{K}}) - L, \vec{\mathcal{K}})$ is a counter–model to $\Gamma \Rightarrow P$. □

Again we extract an algorithm from this proof. It successively picks up each properly nested formula $(\vec{A}_1 \supset Q_1) \supset \ldots \supset (\vec{A}_k \supset Q_k) \supset Q$ where Q is significant, and applies the algorithm recursively to $\vec{A}_i, \Gamma_1, Q_i \supset Q, \Gamma_2 \Rightarrow Q_i$ until all recursive calls are successful. In this case we get a derivation, otherwise we obtain a counter–model. Pruning as described in Sect. 5 is possible here as well.

Since treating the whole fragment would go beyond the limit of space, we have to omit this.

7 Conclusion

We have seen that developing an algorithm by extracting a program from a proof is not only possible, but even easier than proving that a given algorithm is sound and complete. The reason is that we work with the computational content of the rules rather than with their operational semantics. In that way, we presented a new aspect of Hudelmaier's work [7].

8 Implementation

The $\supset$–part of the proof of Theorem 16 is already implemented in the formal system Minlog [10]. In this system, we can use the modified realisability method to

extract an algorithm (see e.g. Troelstra and van Dalen [11]). The present version of MINLOG requires the normalisation of the proof for extracting the algorithm. This means all the recursions are unfolded. Since all the case distinctions are implemented as boolean recursions, the program term is very large and unreadable. In future versions this will no longer be the case. Then, it should not take too much effort to implement the whole proofs of Theorem 16 and 25 and to extract competitive algorithms.

Acknowledgements

Thanks for helpful comments are due to Roy Dyckhoff, Grigori Mints, Karl–Heinz Niggl, Wolfgang Zuber, the anonymous referees and others.

References

1. Roy Dyckhoff. Contraction–free sequent calculi for intuitionistic logic. *Journal of Symbolic Logic*, 57(3):795–807, 1992.
2. Melvin C. Fitting. *Intuitionistic Logic, Model Theory and Forcing*. North-Holland, Amsterdam, 1969.
3. Gerhard Gentzen. Untersuchungen über das logische Schließen. *Math. Zeitschrift*, (39), 1935.
4. Jörg Hudelmaier. Bicomplete calculi for intuitionistical propositional logic. Aufsatz, Universität Tübingen. http://www-pu.informatik.uni-tuebingen.de/logik/joerg/bicomp.ps.
5. Jörg Hudelmaier. A note on kripkean countermodels for intuitionistically unprovable sequents. Aufsatz, Universität Tübingen. http://www-pu.informatik.uni-tuebingen.de/logik/joerg/kripke.ps.Z.
6. Jörg Hudelmaier. A PROLOG program for intuitionistic logic. SNS–Bericht 88–28, Universität Tübingen, 1988.
7. Jörg Hudelmaier. An $O(n \log n)$–space decision procedure for intuitionistic propositional logic. *Journal of Logic and Computation*, 3(1):63–75, 1993.
8. Pierangelo Miglioli, Ugo Moscato, and Mario Ornaghi. How to avoid duplications in refutation systems for intuitionistic logic and Kuroda logic. Rapporto interno 99-93, Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, 1993.
9. Luis Pinto and Roy Dyckhoff. Loop–free construction of counter–models for intuitionistic propositional logic. In Behara, Fritsch, and Lintz, editors, *Symposia Gaussiana, Conf. A*, pages 225–232. De Gruyter, 1995.
10. Helmut Schwichtenberg. *Minlog–an interactive proof system.* http://www.mathematik.uni-muenchen.de/∼logik/minlog_e.html.
11. Anne S. Troelstra and Dirk van Dalen. *Constructivism in Mathematics–an Introduction*, volume 1. North–Holland, 1988.
12. Judith Underwood. A constructive completeness proof for intuitionistic propositional calculus. Technical Report 90-1179, Cornell University, 1990. Also in *Proceedings of the Second Workshop on Theorem Proving with Analytic Tableaux and Related Methods*, April 1993, Marseille, France.
13. N. N. Vorob'ev. A new algorithm for derivability in the constructive propositional calculus. *Amer. Math. Soc. Transl.*, 94(2):37–71, 1970.