A Constructive Completeness Proof for Intuitionistic Propositional Calculus

Judith Underwood*

TR 90-1179
December 1990

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*Supported by a National Science Foundation Graduate Fellowship.

A Constructive Completeness Proof for Intuitionistic Propositional Calculus

Judith Underwood*

December 18, 1990

Abstract

This paper presents a constructive proof of completeness of Kripke models for the intuitionistic propositional calculus. The computational content of the proof is a form of the tableau decision procedure. If a formula is valid, the algorithm produces a proof of the formula in the form of an inhabitant of the corresponding type; if not, it produces a Kripke model and a state in the model such that the formula is not forced at that state in that model.

1 Introduction

Intuitionistic logic has long been known to have important connections with computer science. The Curry-Howard isomorphism describes the strong correspondence between propositional logic and typed lambda calculus – propositions are interpreted as types and proofs of propositions are lambda terms of the corresponding type. This duality gives rise to type theories which have many applications, such as formalizing proofs in systems like Nuprl [1].

A good introduction to intuitionistic logic is Nerode [5], which describes the tableau algorithm in detail. Girard [4] discusses the Curry-Howard isomorphism. The proof is modelled after the completeness proof in Fitting

*Supported by a National Science Foundation Graduate Fellowship.

[3]. A constructive completeness proof for classical propositional logic appears in Constable and Howe [2], which also contains an introduction to Nuprl.

Much work in constructive mathematics has been concerned with the extraction of algorithms from proofs. The proof presented here was developed by the opposite approach; the constructive proof was inspired by the tableau algorithm. One interesting result of this approach is the insight provided into the connection between Kripke models and intuitionistic proofs. If the formula is unprovable, a model which does not force the formula is created from the failed tableau proof; in a sense, they have the same "data structure".

In addition, the description of tableau algorithm which arises from the proof is easy to understand, and in particular (and unlike other presentations) it is very easy to see that the algorithm terminates. The algorithm has double exponential time complexity; the problem is known to be Pspace-complete [6].

One purpose of this paper is to show that the completeness proof is suitable for formalization in Nuprl; hence, definitions will be in the notation of type theory, and the propositional logic proof of the formula (if one exists) will be constructed as an inhabitant of the type corresponding to the proposition. The mathematical and logical equivalents should be clear.

2 Definitions

The first problem is to translate the mathematical definition of validity to type theory. We choose the most general definition of Kripke model, and modify only slightly the definition of forcing.

In mathematical terms, a Kripke model is a triple consisting of a set, a transitive and reflexive relation on that set, and a monotone function from the set to the set of atomic formulas. The set can be thought of as "states of knowledge"; the order relation then describes increasing information, as the set of atomic formulas associated with each state is considered to be the atomic formulas known to be true at that state.

Although Nuprl type theory has set types, it is simpler in this case to consider the set of states of knowledge as a type itself. This is expressed by giving it the type U_1, the lowest in a hierarchy of universe types, that is, types whose members are types. See [1] for more information. The type theoretic definition of Kripke model is then:

$$\begin{aligned} Kripke\_model \;=\;& T : U_1 \\ & \#R : transitive\_reflexive\_relation(T,T) \\ & \#af : monotone\_relation(T, Atomic\_formulas) \end{aligned}$$

Forcing in a particular Kripke model $K = (T, R, af)$ is inductively defined as a predicate $forces(K, s, \alpha)$, where $s \in T$ and $\alpha \in Term$. At the same time (i.e. by mutual recursion), to avoid troublesome negations in the definition of forcing, a predicate $notforces(K, s, \alpha)$ is also defined.

$$\begin{aligned} forces(K,s,\alpha) \;=\;& af(s,\alpha) && (\alpha \in Atom) \\ & forces(K,s,\beta) \wedge forces(K,s,\gamma) && (\alpha = \beta \wedge \gamma) \\ & forces(K,s,\beta) \vee forces(K,s,\gamma) && (\alpha = \beta \vee \gamma) \\ & \forall s'.sRs' \rightarrow && \\ & notforces(K,s',\beta) \vee forces(K,s',\gamma) && (\alpha = \beta \rightarrow \gamma) \end{aligned}$$

$$\begin{aligned} notforces(K,s,\alpha) \;=\;& \neg af(s,\alpha) && (\alpha \in Atom) \\ & notforces(K,s,\beta) \vee notforces(K,s,\gamma) && (\alpha = \beta \wedge \gamma) \\ & notforces(K,s,\beta) \wedge notforces(K,s,\gamma) && (\alpha = \beta \vee \gamma) \\ & \exists s'.sRs' \wedge && \\ & forces(K,s',\beta) \wedge notforces(K,s',\gamma) && (\alpha = \beta \rightarrow \gamma) \end{aligned}$$

Let $Term$ be the type of propositional formulas. As in [2], the proof is described in terms of *sequents.*

$$Sequent = (Term\ list)\#(Term\ list)$$

However, more information is required for the intuitionistic case; thus our basic unit will be a system of sequents.

$$System = (Sequent\ list)$$

(The list type is a convenience; sequents should be thought of as pairs of sets and a system of sequents should be considered a set of sequents.) A given sequent will be written as a pair $< S_T, S_F >$. The formulas in S_T are the ones which are tagged "true" at some node in the tableau algorithm; similarly the formulas in S_F are tagged "false". The sequents themselves correspond to the nodes in the tableau algorithm; they will also correspond to the states in the Kripke model disproving validity, if one exists.

3 Statement of the Theorem

Although we are generally interested only in the validity of a single formula φ, or rather a given sequent $< \emptyset, \varphi >$, the induction requires a statement which is true for systems of sequents. We will prove that, for any system, either there is a sequent in this system which is provable or there is a Kripke model which, for all sequents $< S_T, S_F >$ in the system, forces the formulas in S_T but not those in S_F. This Kripke model will be referred to as a countermodel for the system.

A sequent $< S_T, S_F >$ is provable if there is an inhabitant of the type corresponding to $\bigwedge S_T \rightarrow \bigvee S_F$. (To make the paper accessible to readers without intimate knowledge of type theory, the notation for propositions and types will be the same.)

If there is a countermodel, we also supply a function f from the system to the model such that $f(< S_T, S_F >)$ is the state in the system which forces the formulas in S_T but not those in S_F.

The formal statement of the theorem is then

$$
\begin{array}{l}
\forall S : System. \\
\quad (\exists < S'_T, S'_F > \in S.(\bigwedge S'_T \rightarrow \bigvee S'_F)) \\
\vee \\
(\exists K = (T, R, af) : Kripke\_model. \\
\quad \exists f : S \rightarrow T. \\
\qquad \forall < S_T, S_F > \in S.(f(< S_T, S_F >) = s \rightarrow \\
\qquad (\forall \alpha \in S_T.forces(K, s, \alpha) \wedge \forall \alpha \in S_F.notforces(K, s, \alpha))))
\end{array}
$$

Note that when the theorem is applied to a formula φ, S will contain exactly one sequent, namely $< \emptyset, \varphi >$; thus we have either φ is provable or there is a countermodel which does not force φ.

4 Proof of the Theorem

The proof is by induction on the number of formulas which can be added to the system. Each step either adds a formula to a sequent in S or adds another sequent to S. All formulas added will be subformulas of the formulas already in S, hence only finitely many can be added to a particular sequent. A new sequent $< S_T, S_F >$ is added only when no more formulas can be added to sequents already in S, and then only if there is no sequent $< S'_T, S'_F >$ in S with $S_T \subseteq S'_T$ and $S_F \subseteq S'_F$, so that any new sequent will never develop into a sequent already present in S. Also, the formulas in $< S_T, S_F >$ will be subformulas of the formulas already in S, so only finitely many new sequents can be added.

The proof proceeds by cases.

1. For any pair $< S_T, S_F > \in S$, if $\alpha \wedge \beta \in S_T$ and ($\alpha \notin S_T$ or $\beta \notin S_T$), apply the inductive hypothesis to S', where S' is S with $< S_T, S_F >$ replaced by $< S_T \cup \{\alpha, \beta\}, S_F >$.

   If, for the system S', it is the case that $\alpha \wedge \beta \wedge (\bigwedge S_T) \rightarrow \bigvee S_F$, i.e. we have an inhabitant $f' = \lambda x_1^{\alpha} x_2^{\beta} \ldots x_k^{\alpha \wedge \beta} \ldots$. of the corresponding function type, we can produce an inhabitant f of $\bigwedge S_T \rightarrow \bigvee S_F$. Given an element a of type $\bigwedge S_T$, set f equal to $f'(\pi_1(x_k^{\alpha \wedge \beta}), \pi_2(x_k^{\alpha \wedge \beta}), a)$. Then, since $\alpha \wedge \beta \in S_T$, $f \in \bigwedge S_T \rightarrow \bigvee S_F$. (If some other sequent in S' is the one corresponding to the inhabited type, since it is also in S the theorem still holds.)

   If we have a countermodel $K = (T, R, af)$ for S', and we have $f : S' \rightarrow T$ such that $f(< S_T \cup \{\alpha, \beta\}, S_F >) = s$ for some $s \in T$ (so that $\forall \gamma \in S_T \cup \{\alpha, \beta\}.forces(K, s, \gamma)$), then we trivially have $\forall \gamma \in S_T.forces(K, s, \gamma)$. Letting $f(< S_T, S_F >) = s$, we then have that (T, R, af) is a Kripke countermodel for S.

2. For any pair $< S_T, S_F > \in S$, if $\alpha \wedge \beta \in S_F$ and ($\alpha \notin S_F$ and $\beta \notin S_F$), apply the inductive hypothesis to S_1 and S_2, where S_1 is S with $< S_T, S_F >$ replaced by $< S_T, S_F \cup \{\alpha\} >$ and S_2 is S with $< S_T, S_F >$ replaced by $< S_T, S_F \cup \{\beta\} >$.

   If, for S_1 and S_2, it is the case that $\bigwedge S_T \rightarrow \bigvee S_F \vee \alpha$ and $\bigwedge S_T \rightarrow \bigvee S_F \vee \beta$, so we have functions f_1 and f_2 from the product type (corresponding to) $\bigwedge S_T$ to the disjoint union types (corresponding to) $\bigvee S_F \vee \alpha$ and $\bigvee S_F \vee \beta$. (While $\bigvee S_F$ is really a disjoint union type and should have its elements tagged to identify which disjunct they inhabit, the details of the tagging will be omitted to avoid the extra notation.) To produce a function f to $\bigvee S_F$ given an element a of type $\bigwedge S_T$, decide if $f_1(a) \in \bigvee S_F$. If so, $f(a) = f_1(a)$. Otherwise, if $f_2(a) \in \bigvee S_F$, $f(a) = f_2(a)$. If neither case holds, we must have $f_1(a) \in \alpha$ and $f_2(a) \in \beta$. Then set $f(a) = f_1(a) \# f_2(a)$, an element of $\alpha \wedge \beta$. Since $\alpha \wedge \beta \in S_F$, f has type $\bigwedge S_T \rightarrow \bigvee S_F$.

Otherwise, we have a countermodel $K = (T, R, af)$ for S_1 or S_2. Assume without loss of generality that we have a countermodel for S_1. If we have $f : S_1 \rightarrow T$ such that $f(< S_T, S_F \cup \{\alpha\} >) = s$ for some $s \in T$ (so that $\forall \gamma \in S_F \cup \{\alpha\}.notforces(K, s, \gamma)$), then we trivially have $\forall \gamma \in S_F.notforces(K, s, \gamma)$. Letting $f(< S_T, S_F >) = s$, we then have that (T, R, af) is a Kripke countermodel for S.

3. For any pair $< S_T, S_F > \in S$, if $\alpha \vee \beta \in S_T$ and ($\alpha \notin S_T$ and $\beta \notin S_T$), apply the inductive hypothesis to S_1 and S_2, where S_1 is S with $< S_T, S_F >$ replaced by $< S_T \cup \{\alpha\}, S_F >$ and S_2 is S with $< S_T, S_F >$ replaced by $< S_T \cup \{\beta\}, S_F >$.

   If we have inhabitants f_1 of $\alpha \wedge (\bigwedge S_T) \rightarrow \bigvee S_F$ and f_2 of $\beta \wedge (\bigwedge S_T) \rightarrow \bigvee S_F$, we can produce an element of $\bigwedge S_T \rightarrow \bigvee S_F$. Since $\alpha \vee \beta$ corresponds to a disjoint union type, its inhabitants are tagged so that it is possible to decide whether a given element of $\alpha \vee \beta$ is in α or in β. Given a of type $\bigwedge S_T$, decide whether $\pi_i(a)$ (where i is the index of the element of $\bigwedge S_T$ of type $\alpha \vee \beta$) is in α or β. If $\pi_i(a) \in \alpha$, let $f(a) = f_1(\pi_i(a), a)$; if $\pi_i(a) \in \beta$, let $f(a) = f_2(\pi_i(a), a)$. In algorithmic terms, f could be written "if $\pi_i(a) \in \alpha$ then $f_1(\pi_i(a), a)$ else $f_2(\pi_i(a), a)$". [1] Then f has type $\bigwedge S_T \rightarrow \bigvee S_F$.

   Otherwise, we have a countermodel $K = (T, R, af)$ for S_1 or S_2. Assume without loss of generality that we have a countermodel for S_1. If we have $f : S_1 \rightarrow T$ such that $f(< S_T \cup \{\alpha\}, S_F >) = s$ for some $s \in T$ (so that $\forall \gamma \in S_T \cup \{\alpha\}.forces(K, s, \gamma)$), then we trivially have $\forall \gamma \in S_T.forces(K, s, \gamma)$. Letting $f(< S_T, S_F >) = s$, we then have that (T, R, af) is a Kripke countermodel for S.

4. For any pair $< S_T, S_F > \in S$, if $\alpha \vee \beta \in S_F$ and ($\alpha \notin S_F$ or $\beta \notin S_F$), apply the inductive hypothesis to S', where S' is S with $< S_T, S_F >$ replaced by $< S_T, S_F \cup \{\alpha, \beta\} >$.

   If we have an inhabitant f' of $\bigwedge S_T \rightarrow \bigvee S_F \vee \alpha \vee \beta$, we can produce an inhabitant f of $\bigwedge S_T \rightarrow \bigvee S_F$. Given a of type $\bigwedge S_T$, decide if $f'(a)$

[1] In Nuprl type theory, this would be written $decide(\pi_i(a); x.f_1(x, a), x.f_2(x, a))$.

is in $\bigvee S_F$. If so, let $f(a) = f'(a)$. Otherwise, $f'(a) \in \alpha$ or $f'(a) \in \beta$. If $f'(a) \in \alpha$, let $f(a)$ be $f'(a)$, tagged so that it is an element of the disjoint union $\alpha \vee \beta$. Similarly, if $f'(a) \in \beta$, let $f(a)$ be $f'(a)$, tagged as an element of $\alpha \vee \beta$. Then, since $\alpha \vee \beta \in S_F$, f has type $\bigwedge S_T \to \bigvee S_F$.

Otherwise, we have a countermodel $K = (T, R, af)$ for S'. If we have $f : S' \to T$ such that $f(< S_T, S_F \cup \{\alpha, \beta\} >) = s$ for some $s \in T$ (so that $\forall \gamma \in S_F \cup \{\alpha, \beta\}.notforces(K, s, \gamma)$), then we trivially have $\forall \gamma \in S_F.notforces(K, s, \gamma)$. Letting $f(< S_T, S_F >) = s$, we then have that (T, R, af) is a Kripke countermodel for S.

5. For any pair $< S_T, S_F > \in S$, if $\alpha \to \beta \in S_T$ and ($\alpha \notin S_F$ or $\beta \notin S_T$), apply the inductive hypothesis to S_1 and S_2, where S_1 is S with $< S_T, S_F >$ replaced by $< S_T, S_F \cup \{\alpha\} >$ and S_2 is S with $< S_T, S_F >$ replaced by $< S_T \cup \{\beta\}, S_F >$.

   If, for S_1 and S_2, it is the case that $\bigwedge S_T \to \bigvee S_F \vee \alpha$ and $\bigwedge S_T \wedge \beta \to \bigvee S_F$, then we have a function f_1 from $\bigwedge S_T$ to $\bigvee S_F \vee \alpha$ and a function f_2 from $\beta \wedge \bigwedge S_T$ to $\bigvee S_F$. To produce a function f from $\bigwedge S_T$ to $\bigvee S_F$, given an element a of type $\bigwedge S_T$, decide if $f_1(a) \in \bigvee S_F$. If so, $f(a) = f_1(a)$. Otherwise, $f_1(a) \in \alpha$. Now since $\alpha \to \beta \in S_T$, we can set $f(a)$ equal to $f_2(x^{\alpha \to \beta} f_1(a), a)$ (since $x^{\alpha \to \beta} f_1(a)$ has type β). Then f is of type $\bigwedge S_T \to \bigvee S_F$.

   Otherwise, we have a countermodel $K = (T, R, af)$ for S_1 or S_2. If K is a countermodel for S_1, and $f : S_1 \to T$ such that $f(< S_T, S_F \cup \{\alpha\} >) = s$ for some $s \in T$ (so that $\forall \gamma \in S_F \cup \{\alpha\}.notforces(K, s, \gamma)$), then we trivially have $\forall \gamma \in S_F.notforces(K, s, \gamma)$. Letting $f(< S_T, S_F >) = s$, we then have that (T, R, af) is a Kripke countermodel for S.

   If we have a countermodel for S_2, and $f : S_2 \to T$ such that $f(< S_T \cup \{\beta\}, S_F >) = s$ for some $s \in T$ (so that $\forall \gamma \in S_T \cup \{\beta\}.forces(K, s, \gamma)$), then we trivially have $\forall \gamma \in S_T.forces(K, s, \gamma)$. Letting $f(< S_T, S_F >) = s$, we then have that (T, R, af) is a Kripke countermodel for S.

6. For any pair $< S_T, S_F > \in S$, if $\alpha \to \beta \in S_F$ and $\forall < S'_T, S'_F > \in S.(S_T \cup \{\alpha\} \not\subseteq S'_T \vee \beta \notin S'_F)$, and none of the previous cases hold, apply the inductive hypothesis to S', where S' is $S \cup \{< S_T \cup \{\alpha\}, \{\beta\} >\}$.

   If, for S', we have an inhabitant f' of $(\bigwedge S_T \wedge \alpha) \to \beta$, then the function $\lambda y^{\wedge S_T}.\lambda x^{\alpha}.f'(y, x)$ is in $\bigwedge S_T \to (\alpha \to \beta)$, so the appropriate tagged version of this function is in $\bigwedge S_T \rightarrow \bigvee S_F$.

If we have a countermodel $K = (T, R, af)$ for S', then since $S \subseteq S'$, $< S_T, S_F > \in S$, so $f(< S_T, S_F >) = s$ for some $s \in T$. This means we already know that $\forall \gamma \in S_T.forces(K, s, \gamma) \wedge \forall \gamma \in S_F.notforces(K, s, \gamma)$. So, we need not change f for S, except to remove $< S_T \cup \{\alpha\}, \{\beta\} >$ from its domain. We then have that (T, R, af) is a Kripke countermodel for S.

7. If none of the above cases hold, we have reached the base case.

Note that because none of the above cases hold, we know that, for every $< S_T, S_F > \in S$,

$$\begin{array}{rcl} \alpha \wedge \beta \in S_T & \rightarrow & \alpha \in S_T \textit{ and } \beta \in S_T \\ \alpha \vee \beta \in S_T & \rightarrow & \alpha \in S_T \textit{ or } \beta \in S_T \\ \alpha \rightarrow \beta \in S_T & \rightarrow & \alpha \in S_F \textit{ or } \beta \in S_T \\ \alpha \wedge \beta \in S_F & \rightarrow & \alpha \in S_F \textit{ or } \beta \in S_F \\ \alpha \vee \beta \in S_F & \rightarrow & \alpha \in S_F \textit{ and } \beta \in S_F \\ \alpha \rightarrow \beta \in S_F & \rightarrow & \exists < S'_T, S'_F > .S_T \subseteq S'_T \textit{ and } \alpha \in S_F \textit{ and } \beta \in S_F \end{array}$$

If there is an $< S_T, S_F > \in S$ such that $\exists \alpha \in S_T \cap S_F$, then $\bigwedge S_T \rightarrow \bigvee S_F$ is inhabited by $\lambda x^{\wedge S_T}.(j, \pi_i(x))$, where i is the index of α in $\bigwedge S_T$ and j is the tag for α in the disjoint union $\bigvee S_F$.

Otherwise, $\forall < S_T, S_F > \in S$, $S_T \cap S_F = \emptyset$. Let $T = S$, and define R, a relation on $T \# T$, by

$$R(< S_T, S_F >, < S'_T, S'_F >) \Leftrightarrow S_T \subseteq S'_T$$

It is easy to see that R is transitive and reflexive. Define af, a relation on $T \# Atomic\_formulas$ by

$$af(< S_T, S_F >, \alpha) \Leftrightarrow \alpha \in S_T.$$

Since af is monotone with respect to R, $(T, R, af) : Kripke\_model$. Note that since $S_T \cap S_F = \emptyset, \alpha \in S_F \rightarrow \neg af(s, \alpha)$.

Define $f : S \to T$ to be the identity map from S to T. To complete the theorem, we must show that

$$\begin{array}{l} \forall < S_T, S_F > \in S. \\ \quad f(< S_T, S_F >) = s \to \\ \qquad (\forall \alpha \in S_T.forces(K, s, \alpha) \wedge \forall \alpha \in S_F.notforces(K, s, \alpha)). \end{array}$$

Since *forces* and *notforces* are defined by mutual induction on the complexity of the formula, the proofs of $(\forall \alpha \in S_T.forces(K, s, \alpha) \wedge \forall \alpha \in S_F.notforces(K, s, \alpha))$ must proceed similarly. For simplicity, however, they will be presented sequentially.

We first prove $\forall \alpha \in S_T.forces(K, s, \alpha)$, by induction on the complexity of α.

(a) α is an atom. Then $forces(K, s, \alpha) = af(s, \alpha)$, true when $\alpha \in S_T$ by definition of af.

(b) $\alpha = \beta \wedge \gamma$. We know $\beta \in S_T$ and $\gamma \in S_T$, so by induction hypothesis $forces(K, s, \beta)$ and $forces(K, s, \gamma)$. Therefore, $forces(K, s, \beta \wedge \gamma)$.

(c) $\alpha = \beta \vee \gamma$. We know $\beta \in S_T$ or $\gamma \in S_T$, so by induction hypothesis $forces(K, s, \beta)$ or $forces(K, s, \gamma)$. Therefore, $forces(K, s, \beta \vee \gamma)$.

(d) $\alpha = \beta \to \gamma$. We need that $\forall s'.sRs' \to notforces(K, s', \beta) \vee forces(K, s', \gamma)$. Now, for all $< S'_T, S'_F >$ such that $f(< S'_T, S'_F >) = s'$, we know $sRs' \Leftrightarrow S_T \subseteq S'_T$, so $\beta \to \gamma \in S'_T$. We also know, then, that $\gamma \in S'_T$ or $\beta \in S'_F$, so $forces(K, s', \gamma)$ or $notforces(K, s', \beta)$. Therefore, $forces(K, s', \beta \to \gamma)$.

The proof for $\forall \alpha \in S_F.notforces(K, s, \alpha)$ is similar.

(a) α is an atom. Since $S_T \cap S_F = \emptyset$, $\alpha \notin S_T$, so $\neg af(s, \alpha)$, i.e. $notforces(K, s, \alpha)$.

(b) $\alpha = \beta \wedge \gamma$. We know $\beta \in S_F$ or $\gamma \in S_F$, so by induction hypothesis $notforces(K, s, \beta)$ or $notforces(K, s, \gamma)$. Therefore, $notforces(K, s, \beta \wedge \gamma)$.

(c) $\alpha = \beta \vee \gamma$. We know $\beta \in S_F$ and $\gamma \in S_F$, so by induction hypothesis $notforces(K, s, \beta)$ and $notforces(K, s, \gamma)$. Therefore, $notforces(K, s, \beta \vee \gamma)$.

(d) $\alpha = \beta \rightarrow \gamma$.
Since $< S_T, S_F > \in T$ and rule 6 above doesn't apply, we must have $\exists < S'_T, S'_F > \in T. S_T \cup \{\beta\} \subseteq S'_T \wedge \gamma \in S'_F$. Let $s' = f(< S'_T, S'_F >)$. By induction hypothesis, $forces(K, s', \beta)$ and $notforces(K, s', \gamma)$. Also, since $S_T \subseteq S_T \cup \{\alpha\} \subseteq S'_T$, sRs'. Therefore, $\exists s'.sRs' \wedge forces(K, s', \beta) \wedge notforces(K, s', \gamma)$.

5 Conclusion

Each system of sequents in the above proof corresponds to a branch of the tableau. If all branches close (i.e. if there is a sequent in each system such that $S_T \cap S_F$ is nonempty), then the formula is provable; else, the open branch becomes a Kripke model which does not force the formula. Note that a state (sequent) is added to the Kripke model only when necessary to provide a state which fails to force an implication; hence, the model produced is minimal. The model produced is finite, so the proof also demonstrates the completeness of finite Kripke models.

If the proof were added to the Nuprl system, the result would be a procedure for deciding the (intuitionistic) validity of a propositional formula. However, from inside Nuprl, we do not yet know that a proposition cannot have both a proof and a countermodel; we have not proved *soundness* of Kripke models. This leads to some interesting questions about what model makes the most sense in type theory; ideally, the soundness and completeness theorems would say something meaningful about the type theory as well as the logic. Such a result would provide deeper understanding of the relationship between logic and type theory, and would give insight into the use of type theory to describe its own semantics (i.e. reflection).

References

[1] R. L. Constable *et al. Implementing Mathematics with the Nuprl Proof Development System.* Prentice-Hall, 1986.

[2] R. L. Constable and D. J. Howe. Implementing Metamathematics as an Approach to Automatic Theorem Proving. In *Formal Techniques in Artificial Intelligence*, R. Banerji, ed. North-Holland, 1990.

[3] M. Fitting. *Intuitionistic Logic, Model Theory, and Forcing.* North-Holland, 1969.

[4] J. Y. Girard. *Proofs and Types.* Cambridge University Press, 1989.

[5] A. Nerode. Some Lectures on Intuitionistic Logic. In *Logic and Computer Science*, Springer-Verlag Lecture Notes in Mathematics 1429, 1990.

[6] R. Statman. Intuitionistic Propositional Logic is Polynomial-space Complete. Theoretical Computer Science 9, 67-72, 1979.