Proof Translations in Classical and Intuitionistic Logic

Ian Ribeiro[1], oriented by Olivier Hermant

MINES ParisTech, PSL Research University, France
ian.ribeiro_de_faria_leite@minesparis.psl.eu

Abstract. This work aims to discuss the translation between Tableaux proofs and sequent calculus proofs in first-order intuitionistic propositional logic and in first-order classical propositional logic. It begins with an overview of the definitions and clarification of the notations. It then shows a translation process in classical logic and its OCAML implementation. Finally, a potential extension towards translation in intuitionistic logic is discussed.

Keywords: Tableaux proof · sequent calculus · intuitionistic logic.

1 Introduction

1.1 The topic

Sequent calculus and the tableaux Method are both fundamental systems for representing and systematically generating logical proofs. Sequent calculus highlights the structure and rules of deduction, while tableaux proofs provide a more intuitive approach. Translating between proof systems is important for automatically verifying automatically generated proofs and understanding their relationships.

In this work, sentences will implicitly refer to first-order predicate logic sentences; In the translation discussed in chapter V, they will be implicitly restricted to the propositional fragment of first order logic in order to simplify the discussion. For intuitionistic logic, their meaning will come from Kripke's semantics [3] . The notation for structures and frames, as well as the motivation for the intuitionistic tableaux method will be heavily based on [1]. As in most of the proofs for the intuitionistic tableaux from [1], there wont be any functions. In order to make this document slightly more self-contained and to clarify the notation, we will briefly explain:

1.2 Definitions

Definition 1. ***A Structure of a Language*** *(an $\mathcal{L}$-structure) is a triple that consists of a domain and two assignments:*

- *An assignment from the constant symbols of the language to the domain.*

- *An assignment from the predicate symbols of the language to predicates in the domain.*

Here the domain is the finite set of the objects being described by the language.

Structures represent possible worlds or possible states of knowledge inside a frame:

Definition 2. ***A Kripke Frame*** *of a Language $\mathcal{L}$, $\mathcal{C} = (R, \{C(p)\}_{p \in R})$ consists of a partially ordered set R, and an $\mathcal{L}$-structure $C(p)$ for all p's in R. Furthermore, in a Kripke Frame, if $p \leq q$, then $C(q)$ extends $C(p)$ in the following sense:*

- *All sentences that are true in $C(p)$ are true in $C(q)$.*
- *The domain of $C(p)$ is included in the domain of $C(q)$.*
- *The assignments in $C(p)$ are the same as in $C(q)$ for the domain of $C(p)$*

Particularly, in order to simplify the notation, R will always be a set of sequences of integers in lexicographic order: $p \leq q$ if it exists an l such that $q = p : l$. Also, from now on, the constant elements of a language will always be in an ordered set $\{c_0, c_1, c_2 ...\}$

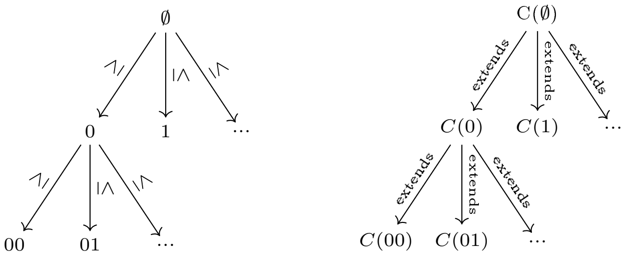

Fig. 1. R and a Kripke frame.

Definition 3. ***Forcing****. When a sentence ϕ of a language $\mathcal{L}$ is* ***forced*** *by a structure $C(p)$ of a $\mathcal{L}$-frame $\mathcal{C}$, we denote it as: $p \Vdash_{\mathcal{C}} \phi$*
Forcing is defined by induction: [1]

- *$p \Vdash_{\mathcal{C}} \phi$ iff ϕ is true in $C(p)$ (if ϕ is an atomic sentence).*
- *$p \Vdash_{\mathcal{C}} (\phi \to \psi)$ iff for all $q \geq p$, if $q \Vdash_{\mathcal{C}} \phi$, then $q \Vdash_{\mathcal{C}} \psi$.*
- *$p \Vdash_{\mathcal{C}} \neg\phi$ iff for all $q \geq p$, q does not force ϕ.*
- *$p \Vdash_{\mathcal{C}} (\forall x)\phi(x)$ iff for all $q \geq p$ and d in $\mathcal{L}_{C(q)}$, $q \Vdash_{\mathcal{C}} \phi(d)$.*
- *$p \Vdash_{\mathcal{C}} (\exists x)\phi(x)$ iff there exists a d in $\mathcal{L}_{C(q)}$, such that $p \Vdash_{\mathcal{C}} \phi(d)$.*
- *$p \Vdash_{\mathcal{C}} (\phi \wedge \psi)$ iff $p \Vdash_{\mathcal{C}} \phi$ and $p \Vdash_{\mathcal{C}} \psi$.*
- *$p \Vdash_{\mathcal{C}} (\phi \vee \psi)$ iff $p \Vdash_{\mathcal{C}} \phi$ or $p \Vdash_{\mathcal{C}} \psi$.*

By $\mathcal{L}_{C(q)}$ we are implicitly talking about the language of the structure $C(q)$. [1] defines language extensions with more details that are not necessary here.

Definition 4. ***Intuitionistic Validity***. *A sentence of a language $\mathcal{L}$ is Intuitionistically valid if it is forced in all structures of all Kripke frames of $\mathcal{L}$.*

In classical logic, this definition simplifies, and it is simplified again by the fact that there is a unique world (R is a singleton); in fact, here we will define classical validity as: [1]

Definition 5. ***Classical Validity***. *A sentence of a language $\mathcal{L}$ is classically valid if it is forced by all singleton structure Kripke frames of $\mathcal{L}$.*

2 The Intuitionistic Tableaux Method

Considerations Here we first define a slightly different version of the destructive tableaux proof tree described by [1], where each node is a truth assertion. This different version will allow for a more implementation-oriented approach and the translation later on.

The correspondence of the destructive tableaux proof tree described in [1] to our new one is shown in Figure 2. Generally speaking, a node collects all the signed sentences that appeear on the sequence of nodes in the path that goes from the root to it. Afterwards, some nodes are removed from the newly formed tableau by adjoining its children and its parent. A node should be removed if its corresponding node in the original tableau was not a leaf of the atomic tableau rule [1] that introduced it.

In this new format, one could see each node of the tree as an assumption of the existence of a frame that respects a list of constraints, and each edge as an implication between assumptions.

The Intuitionistic Tableaux Method The tableaux method stands on some definitions, they will be justified briefly:

Definition 6. ***A Signed Sentence*** *is a forcing assertion inside of a tableaux proof. It has the format $T_q\phi$ or $F_p\phi$*
A Signed Sentence List *is a list of forcing assertions inside of a tableaux proof. We say that a list of forcing assertions L made out of sentences $\{T_{p_1}\gamma_1, T_{p_2}\gamma_2, ...\}$ and $\{F_{q_1}\delta_1, F_{q_2}\delta_2 ...\}$ is intuitionistically valid if there exists a frame $\mathcal{C}$ that respects L :*
$\mathcal{C}(p_1) \Vdash \gamma_1$ *and* $\mathcal{C}(p_2) \Vdash \gamma_2$ *and* ... $\mathcal{C}(q_1) \nVdash \delta_1$ *and* $\mathcal{C}(q_2) \nVdash \delta_2$

A signed sentence list can be seen as a model-existence assumption that may or may not consistently lead to a frame. We can infer other assumptions from a signed sentence list. The function f, defined below, is one of the ways we can do that:

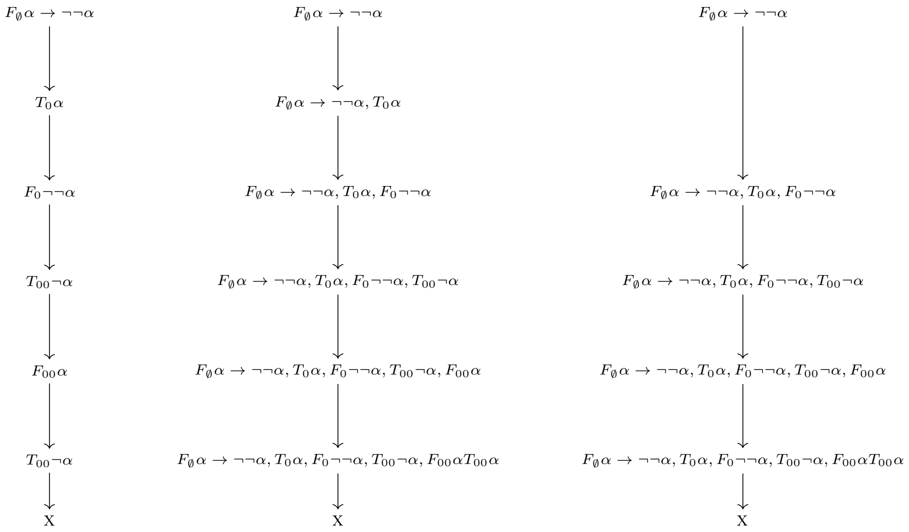
$F_\emptyset \alpha \to \neg\neg\alpha$
$T_0\alpha$
$F_0\neg\neg\alpha$
$T_{00}\neg\alpha$
$F_{00}\alpha$
$T_{00}\neg\alpha$
X
$F_\emptyset \alpha \to \neg\neg\alpha$
$F_\emptyset \alpha \to \neg\neg\alpha, T_0\alpha$
$F_\emptyset \alpha \to \neg\neg\alpha, T_0\alpha, F_0\neg\neg\alpha$
$F_\emptyset \alpha \to \neg\neg\alpha, T_0\alpha, F_0\neg\neg\alpha, T_{00}\neg\alpha$
$F_\emptyset \alpha \to \neg\neg\alpha, T_0\alpha, F_0\neg\neg\alpha, T_{00}\neg\alpha, F_{00}\alpha$
$F_\emptyset \alpha \to \neg\neg\alpha, T_0\alpha, F_0\neg\neg\alpha, T_{00}\neg\alpha, F_{00}\alpha T_{00}\alpha$
X
$F_\emptyset \alpha \to \neg\neg\alpha$
$F_\emptyset \alpha \to \neg\neg\alpha, T_0\alpha, F_0\neg\neg\alpha$
$F_\emptyset \alpha \to \neg\neg\alpha, T_0\alpha, F_0\neg\neg\alpha, T_{00}\neg\alpha$
$F_\emptyset \alpha \to \neg\neg\alpha, T_0\alpha, F_0\neg\neg\alpha, T_{00}\neg\alpha, F_{00}\alpha$
$F_\emptyset \alpha \to \neg\neg\alpha, T_0\alpha, F_0\neg\neg\alpha, T_{00}\neg\alpha, F_{00}\alpha T_{00}\alpha$
X

Fig. 2. Example of a destructive tableaux proof tree from [1], the intermediary structure and the non-destructive tableaux proof tree.

Definition 7. *The function f takes a signed sentence σ and a signed sentence list L and returns one or two signed sentence lists.*
$f(\sigma, L)$ is defined as follows:
(here we denote $l : l' = l_1, l_2..., l_{|l|}, l'_1, l'_2.., l'_{|l'|}$)
if $\sigma \in L$:

- $f(T_p\alpha, L) = [L - \sigma : \sigma : F_{p'}\alpha]$
  for a $p' \geq p$ such that $T_{p'}\alpha$ not present in L. (α is atomic)
- $f(T_p\neg\alpha, L) = [L - \sigma : \sigma : F_{p'}\alpha]$
  for a $p' \geq p$ such that $T_{p'}, L$ not present in L.
- $f(F_p\neg\alpha, L) = [L - \sigma : \sigma : T'_p\alpha]$
  for a new $p' \geq p$
- $f(T_p(\alpha \wedge \beta), L) = [L - \sigma : \sigma : T_p\alpha : T_p\beta]$.
- $f(F_p(\alpha \wedge \beta), L) = [L - \sigma : \sigma : F_p\alpha], [L - \sigma : \sigma : F_p\beta]$.
- $f(T_p(\alpha \vee \beta), L) = [L - \sigma : \sigma : T_p\alpha], [L - \sigma : \sigma : T_p\beta]$.
- $f(F_p(\alpha \vee \beta), L) = [L - \sigma : \sigma : F_p\alpha : F_p\beta]$.
- $f(T_p(\alpha \to \beta), L) = [L - \sigma : \sigma : F_{p'}\alpha], [L - \sigma : \sigma : T_{p'}\beta]$
  for a new $p' \geq p$
- $f(F_p(\alpha \to \beta), L) = [L - \sigma : \sigma : T_{p'}\alpha : F_{p'}\beta]$
  for a $p' \geq p$ present in $\sigma : L$ and such that $T_{p'}(\alpha)$ and $F_{p'}(\beta)$ not L.
- $f(T_p(\forall x)\phi(x), L) = [L - \sigma : \sigma : T_p\phi(c_i)]$
  for the first constant c_i such that $T_p\phi(c_i)$ is not in L.
- $f(F_{p'}(\forall x)\phi(x), L) = [L - \sigma : \sigma : F_p\phi(c_i)]$
  for the first constant c_i not present in $\sigma : L$ and a new $p' \geq p$

- $f(T_{p'}(\exists x)\phi(x), L) = [L - \sigma : \sigma : T_p\phi(c_i)]$
  for the first constant c_i not present in $\sigma : L$ and a new $p' \geq p$.
- $f(F_p(\exists x)\phi(x), L) = [L - \sigma : \sigma : F_p\phi(c_i)]$
  for the first constant c_i such that $F_p\phi(c_i)$ is not in L and .

if $\sigma \notin L$:

- $f(\sigma, L) = L$

Although not justified here, the reordering of the terms plays an important role on the implementation and in the completeness of the systematic tableaux method.: each signed sentence will be σ infinitely many times.

On the other hand, the soundness of the tableaux method states that f is well-behaved:

Theorem 1. *Given a signed sentence list L, if L is intuitionistically valid then f(L) is intuitionistically valid*

Proof. [1] proves this by using the definition of forcing. We will do the most complex case, that has the most part of the ideas needed for the other ones:

Case: if $L \ni F_p(\forall x)\phi(x)$ is intuitionistically valid, then there exists a frame $\mathcal{C}$ that respects it (in the sense of the Definition 6). $\mathcal{C}$ contains q and a c $\in \mathcal{C}(q)$ such that $\mathcal{C}(c) \not\Vdash \phi(c)$. Let $\mathcal{C}'$ be the same as $\mathcal{C}$, with the new world $\mathcal{C}(p')$ such that $\mathcal{C}(p') := \mathcal{C}(q)$ and $c_i := c$

In other words, let $\mathcal{C}'$ be a frame that is exactly like $\mathcal{C}$ with addition of a new structure $p' \geq p$ such that $\mathcal{C}'(p') \not\Vdash \phi(c_i)$ for a new c_i. We know that $\mathcal{C}'$ exists by the definition of forcing for $\forall$ in C. Also $\mathcal{C}'$ is a frame that respects $f(\forall x\phi(x), L) = [L - \forall x\phi(x) : \forall x\phi(x) : T_p\phi(c_i)]$.

Definition 8. ***The tableaux Development of a List of Sentences*** *is defined inductively:*

- *A tree with the single node L is a tableaux development of L.*
- *If τ is a tableaux development of L, then $\hookleftarrow (\sigma, \tau)$ is a tableaux development of L. Where:*

$$\hookleftarrow (\sigma, \tau) = \textit{the extension of } \tau \textit{ with } f(\sigma, l)\textit{added to all leaves } l \textit{ that contain } \sigma$$

Theorem 2. *The following implication is true:*
If L is intuitionistically valid then one of the leaves of $\hookleftarrow (\sigma_n, \hookleftarrow (\sigma_{n-1}(.....(\hookleftarrow (\sigma_1, L)...))))$ is intuitionistically valid.

Proof. If L is intuitionistically invalid, the premise is false and the implication holds. If L is intuitionistically valid, The proof goes by induction on the number of times $\hookleftarrow$ is applied :

The base case is true by definition: L is a intuitionistically valid leaf. Next we assume $\tau' = \hookleftarrow (\sigma, \tau)$. By the induction hypothesis, there exists an intuitionistically valid leaf σ in τ.

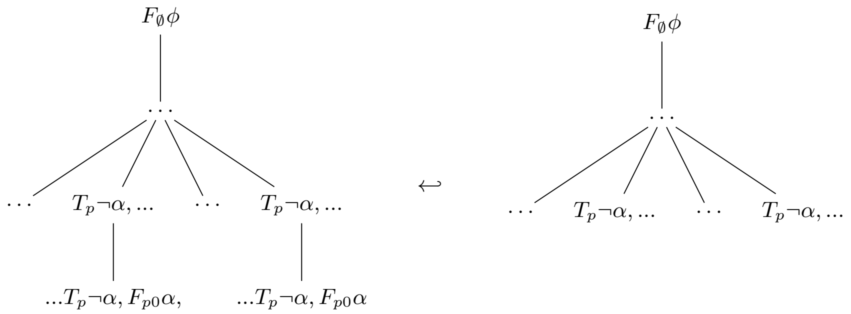

Fig. 3. Example of $\hookleftarrow (T\neg\alpha, \tau)$ and τ.

- If σ is a leaf in τ, then the theorem holds for τ'.
- If σ is not a leaf in τ, then: One or two nodes were added to σ in τ' by the definition of $\hookleftarrow$. By the theorem 1, one of the added nodes is also intuitionistically valid. Consequently, the theorem holds for τ'.

From this, we can conclude that: If all leaves of $\hookleftarrow (\sigma_n, \hookleftarrow (\sigma_{n-1}(.....(\hookleftarrow (\sigma_1, F\phi)...))))$ are contradictory, then $F\phi$ is not intuitionistically valid, by consequence ϕ is intuitionistically valid.

Completeness For purposes of implementation, we define an $\hookleftarrow_c$ that can be applied systematically:

Definition 9. *$\hookleftarrow_c (\tau) = \tau$ with $f(h, l)$ added to all leaves l that contain σ. h is the first signed sentence of the shallowest (and after that leftmost) non-contradictory leaf. here, c stands for complete. A contradictory leaf is a leaf that has T and F of the same sentence.*

We can use $\hookleftarrow_c$ instead of $\hookleftarrow$ to define a systematic tableaux method. This was important for generating tableaux proofs in order to test the translation implementation. This will not be discussed further, as the translation assumes a tableaux proof not necessarily equal to one given by $\hookleftarrow_c$.

3 The Classical Tableaux Method

The classical tableaux method can be seen as the intuitionistic tableaux method restricted to one-framed structures. The adapted definitions are:

Definition 10. ***A Signed Sentence** is a forcing assertion inside of a tableaux proof. It can have the format $T\phi$ or $F\phi$*
***A Signed Sentence List** is a list of forcing assertions inside of a tableaux proof. We say that a list of forcing assertions L made out of sentences $\{T\gamma_1, T\gamma_2, ...\}$ and $\{F\delta_1, F\delta_2 ...\}$ is "classically valid" if there exists a single-structure frame $\mathcal{C}$ such that $\mathcal{C}(\emptyset) \Vdash \gamma_1$ and $\mathcal{C}(\emptyset) \Vdash \gamma_2$ and ... $\mathcal{C}(\emptyset) \nVdash \delta_1$ and $\mathcal{C}(\emptyset) \nVdash \delta_2$*

Definition 11. *The function f takes a signed sentence σ and a signed sentence list L and returns one or two signed sentence lists.*
$f(\sigma, L)$ is defined as follows:
(here we denote $l : l' = l_1, l_2..., l_{|l|}, l'_1, l'_2.., l'_{|l'|}$)
if $\sigma \in L$:

- $f(T\neg\alpha, L) = [L - \sigma : \sigma : F\alpha]$.
- $f(F\neg\alpha, L) = [L - \sigma : \sigma : T\alpha]$.
- $f(T(\alpha \wedge \beta), L) = [L - \sigma : \sigma : T\alpha : T\beta]$.
- $f(F(\alpha \wedge \beta), L) = [L - \sigma : \sigma : F\alpha], [L : \sigma : F\beta]$.
- $f(T(\alpha \vee \beta), L) = [L - \sigma : \sigma : T\alpha], [L : \sigma : T\beta]$.
- $f(F(\alpha \vee \beta), L) = [L - \sigma : \sigma : F\alpha : F\beta]$.
- $f(T(\alpha \to \beta), L) = [L - \sigma : \sigma : F\alpha], [L : \sigma : T\beta]$.
- $f(F(\alpha \to \beta), L) = [L - \sigma : \sigma : T\alpha : F\beta]$.
- $f(T(\forall x)\phi(x), L) = [L - \sigma : \sigma : T\phi(c_i)]$
  for the first constant c_i such that $T\phi(c_i)$ is not in L.
- $f(F(\forall x)\phi(x), L) = [L - \sigma : \sigma : F\phi(c_i)]$
  for the first constant c_i not present in L.
- $f(T(\exists x)\phi(x), L) = [L - \sigma : \sigma : T\phi(c_i)]$
  for the first constant c_i not present in L.
- $f(F(\exists x)\phi(x), L) = [L - \sigma : \sigma : F\phi(c_i)]$
  for the first constant c_i such that $F\phi(c_i)$ is not in L.

if $\sigma \notin L$:

- $f(\sigma, L) = L$

4 Sequent Calculus

Here we use the multi-conclusion sequent calculus defined in [2].

Definition 12. ***A Sequent*** *is an expression of the form*

$$\Gamma \vdash \Delta$$

where $\Gamma = \{\Gamma_1, \Gamma_2, \Gamma_3, ...\}$ and $\Delta = \{\Delta_1, \Delta_2, \Delta_3 ...\}$ are finite sets of formulas. Γ is called the antecedent, and Δ is called the succedent.

Γ represents multiple necessary hypothesis, while Δ represents multiple possible conclusions.

Definition 13. ***(Intuitionistic Validity of a Sequent)*** *A sequent is intuitionistically valid if for all Kripke frames, if all of Γ are forced, then at least one of Δ is forced.*

The more awkward definition:

" A sequent is intuitionistically valid if there does not exist a frame that forces all sentences in Γ and does not force any sentence in Δ. "

when compared with the definition of list of signed sentences, hints us towards an semantical equivalence between signed sentence lists and sequents for single framed structures. This will not be discussed in depth.

A sequent-calculus rule can be interpreted as an implication: if the top sequent is valid, then the bottom sequent is valid. Syntactically, each rule represents vertices that can be added to a sequent proof-tree development to obtain another sequent proof tree development

4.1 Classical Sequent Calculus

Definition 14. *A single sequent is a classical sequent proof-tree development*
Any of the rules of the table 1 applied to a classical sequent proof-tree development is a sequent proof-tree development
A classical sequent proof-tree is a classical sequent proof-tree development with axioms on all leaves.

Rules for classical multi-consequence sequent calculus are given in Table1

Table 1. Rules for classical multi-consequence sequent calculus.

Rule	**Inference**
Axiom	$\dfrac{}{\alpha \vdash \alpha}$
Weakening	$\dfrac{\Gamma \vdash \Delta}{\Gamma, \alpha \vdash \Delta}$ $\quad$ $\dfrac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, \alpha}$
Negation	$\dfrac{\Gamma, \vdash \alpha, \Delta}{\Gamma, \neg\alpha \vdash \Delta}\ \neg\mathrm{R}\ \neg\alpha$ $\quad$ $\dfrac{\Gamma, \alpha \vdash \Delta}{\Gamma \vdash \neg\alpha, \Delta}\ \neg\mathrm{L}\ \neg\alpha$
Conjunction	$\dfrac{\Gamma, \alpha, \beta \vdash \Delta}{\Gamma, \alpha \wedge \beta \vdash \Delta}\ \wedge\mathrm{L}\ \alpha \wedge \beta$ $\quad$ $\dfrac{\Gamma \vdash \alpha, \Delta \qquad \Gamma \vdash \beta, \Delta}{\Gamma \vdash \alpha \wedge \beta, \Delta}\ \wedge\mathrm{R}\ \alpha \wedge \beta$
Disjunction	$\dfrac{\Gamma, \alpha \vdash \Delta \qquad \Gamma, \beta \vdash \Delta}{\Gamma, \alpha \vee \beta \vdash \Delta}\ \vee\ \mathrm{L}\ \alpha \vee \beta$ $\quad$ $\dfrac{\Gamma \vdash \alpha, \Delta}{\Gamma \vdash \alpha \vee \beta, \Delta}\ \vee\ R_1\ \alpha \vee \beta$ $\quad$ $\dfrac{\Gamma \vdash \beta, \Delta}{\Gamma \vdash \alpha \vee \beta, \Delta}\ \vee\ R_2\ \alpha \vee \beta$
Implication	$\dfrac{\Gamma \vdash \alpha, \Delta \qquad \Gamma, \beta \vdash \Delta}{\Gamma, \alpha \rightarrow \beta \vdash \Delta}\ \rightarrow \mathrm{L}\ \alpha \rightarrow \beta$ $\quad$ $\dfrac{\Gamma, \alpha \vdash \beta, \Delta}{\Gamma \vdash \alpha \rightarrow \beta, \Delta}\ \rightarrow \mathrm{R}\ \alpha \rightarrow \beta$
Quantifiers	$\dfrac{\Gamma, \alpha(t) \vdash \Delta}{\Gamma, \forall x\alpha(x) \vdash \Delta}\ \forall\ \mathrm{L}\ \forall x\alpha(x)$ $\quad$ $\dfrac{\Gamma \vdash \alpha(y), \Delta}{\Gamma \vdash \forall x\alpha(x), \Delta}\ \forall\ \mathrm{R}\ \forall x\alpha(x)$ with y fresh $\dfrac{\Gamma, \alpha(y) \vdash \Delta}{\Gamma, \exists x\alpha(x) \vdash \Delta}\ \exists\ \mathrm{L}\ \exists x\alpha(x)$ with y fresh $\quad$ $\dfrac{\Gamma \vdash \alpha(t), \Delta}{\Gamma \vdash \exists x\alpha(x), \Delta}\ \exists\ \mathrm{R}\ \exists x\alpha(x)$
Contraction	$\dfrac{\Gamma, \alpha, \alpha \vdash \Delta}{\Gamma, \alpha \vdash \Delta}$ $\quad$ $\dfrac{\Gamma \vdash \alpha, \alpha, \Delta}{\Gamma \vdash \alpha, \Delta}$

4.2 Intuitionistic Sequent Calculus

Definition 15. *A single sequent is an intuitionistic sequent proof-tree development*
Any of the rules of table 2 applied to an intuitionistic sequent proof-tree development is an intuitionistic sequent proof-tree development
A intuitionistic squent proof-tree is a intuitionistic sequent proof-tree development with axioms on all leafs.

Rules for intuitionistic multi-consequence sequent calculus are given in Table2

Table 2. Rules for intuitionistic multi-consequence sequent calculus.

Rule	**Inference**
Axiom	$\dfrac{}{\alpha \vdash \alpha}$
Weakening	$\dfrac{\Gamma \vdash \Delta}{\Gamma, \alpha \vdash \Delta}$ $\quad$ $\dfrac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, \alpha}$
Negation	$\dfrac{\Gamma, \vdash \alpha, \Delta}{\Gamma, \neg\alpha \vdash \Delta}\ \neg\mathrm{L}\ \neg\alpha$ $\quad$ $\dfrac{\Gamma, \alpha \vdash}{\Gamma \vdash \neg\alpha,}\ \neg\mathrm{R}\ \neg\alpha$
Conjunction	$\dfrac{\Gamma, \alpha, \beta \vdash \Delta}{\Gamma, \alpha \wedge \beta \vdash \Delta}\ \wedge\mathrm{L}\ \alpha \wedge \beta$ $\quad$ $\dfrac{\Gamma \vdash \alpha, \Delta \qquad \Gamma \vdash \beta, \Delta}{\Gamma \vdash \alpha \wedge \beta, \Delta}\ \wedge\mathrm{R}\ \alpha \wedge \beta$
Disjunction	$\dfrac{\Gamma, \alpha \vdash \Delta \qquad \Gamma, \beta \vdash \Delta}{\Gamma, \alpha \vee \beta \vdash \Delta}\ \vee\ \mathrm{L}\ \alpha \vee \beta$ $\quad$ $\dfrac{\Gamma \vdash \alpha, \Delta}{\Gamma \vdash \alpha \vee \beta, \Delta}\ \vee\ R_1\ \alpha \vee \beta$ $\quad$ $\dfrac{\Gamma \vdash \beta, \Delta}{\Gamma \vdash \alpha \vee \beta, \Delta}\ \vee\ R_2\ \alpha \vee \beta$
Implication	$\dfrac{\Gamma \vdash \alpha, \Delta \qquad \Gamma, \beta \vdash \Delta}{\Gamma, \alpha \rightarrow \beta \vdash \Delta}\ \rightarrow\mathrm{L}\ \alpha \rightarrow \beta$ $\quad$ $\dfrac{\Gamma, \alpha \vdash \beta}{\Gamma \vdash \alpha \rightarrow \beta}\ \rightarrow\mathrm{R}\ \alpha \rightarrow \beta$
Quantifiers	$\dfrac{\Gamma, \alpha(t) \vdash \Delta}{\Gamma, \forall x\alpha(x) \vdash \Delta}\ \forall\ \mathrm{L}\ \forall x\alpha(x)$ $\quad$ $\dfrac{\Gamma \vdash \alpha(y)}{\Gamma \vdash \forall x\alpha(x)}\ \forall\ \mathrm{R}\ \forall x\alpha(x)$ with y fresh $\dfrac{\Gamma, \alpha(y) \vdash \Delta}{\Gamma, \exists x\alpha(x) \vdash \Delta}\ \exists\ \mathrm{L}\ \exists x\alpha(x)$ with y fresh $\quad$ $\dfrac{\Gamma \vdash \alpha(t), \Delta}{\Gamma \vdash \exists x\alpha(x), \Delta}\ \exists\ \mathrm{R}\ \exists x\alpha(x)$
Contraction	$\dfrac{\Gamma, \alpha, \alpha \vdash \Delta}{\Gamma, \alpha \vdash \Delta}$ $\quad$ $\dfrac{\Gamma \vdash \alpha, \alpha, \Delta}{\Gamma \vdash \alpha, \Delta}$

5 Translation

Definition 16. ***Node Translating Function*** $\mathcal{T}$

Given a signed sentence list L with sentences $\{T_{p_1}\gamma_1, T_{p_2}\gamma_2, ...\}$ *and* $\{F_{q_1}\delta_1, F_{q_2}\delta_2...\}$ *and a frame* $w \in \{p_1, p_2, ...\} \cup \{q_1, q_2, ...\}$ *then:*
$\mathcal{T}(L, w) = \Gamma_1, \Gamma_2, ... \vdash \Delta_1, \Delta_2, ...$*, where:*
$\{T_{p'_1}\Gamma_1, T_{p'_2}\Gamma_2, ...\}$ *are the elements of* $\{T_{p_1}\gamma_1, T_{p_2}\gamma_2, ...\}$ *such that* $p' \leq w$ *and* $\{F_{q'_1}\Delta_1, F_{q'_2}\Delta_2, ...\}$ *are the elements of* $\{F_{q_1}\delta_1, F_{q_2}\delta_2, ...\}$ *such that* $q' \geq w$
For the classical case: $\mathcal{T}(L) = \mathcal{T}(L, \emptyset)$

Theorem 3. *Given signed sentence list L and a* w*, if L is intuitionistically valid then the sequent* $\mathcal{T}(L, w)$ *is not intuitionistically valid.*

Proof. If L has the sentences $\{T_{p_1}\gamma_1, T_{p_2}\gamma_2, ...\}$ and $\{F_{q_1}\delta_1, F_{q_2}\delta_2...\}$,it exists a frame $\mathcal{C}$ such that:

$\mathcal{C}(p_1) \Vdash \gamma_1$ and $\mathcal{C}(p_2) \Vdash \gamma_2$ and ... $\mathcal{C}(q_1) \nVdash \delta_1$ and $\mathcal{C}(q_2) \nVdash \delta_2$
which implicitly entails, by the definition of extension in frames:

$(\mathcal{C}(p) \Vdash \gamma_1$ for all $p \geq p_1)$ and $(\mathcal{C}(p) \Vdash \gamma_2$ for all $p \geq p_2)$ and ... $(\mathcal{C}(p) \nVdash \delta_1$ for all $p \leq q_1)$ and $(\mathcal{C}(p) \nVdash \delta_2$ for all $p \leq q_2)$...

Take the structure $\mathcal{C}(w)$ inside of $\mathcal{C}$. We can infer:

$\mathcal{C}(w) \Vdash \Gamma_1$ and $\mathcal{C}(w) \Vdash \Gamma_2$ and ... $\mathcal{C}(w) \nVdash \Delta_1$ and $\mathcal{C}(w) \nVdash \Delta_2$....

$\mathcal{C}$ is thus a counterexample proving the non-validity of $\Gamma_1, \Gamma_2, ... \vdash \Delta_1, \Delta_2, ...$

5.1 Classical Translation

Theorem 4. $\begin{array}{c} \mathcal{T}(f(\sigma, L)) \\ \hline\hline \mathcal{T}(l) \end{array}$ *is classically admissible*

Proof. Here will show the implicit contraction rule being used:
If $L - \sigma$ has sentences $\{T\Gamma_1, T\Gamma_2, ...\}$ and $\{F\Delta_1, F\Delta_2...\}$, $\Gamma = \Gamma_1, \Gamma_2$, ... and $\Delta = \Delta_1, \Delta_2$,... then:

- If $f = f_{T\neg}$:

$$\begin{array}{c} \mathcal{T}(f(T\neg\alpha, L)) \\ \hline\hline \mathcal{T}(L) \end{array} = \begin{array}{c} \mathcal{T}((L - T\neg\alpha) : T\neg\alpha : F\alpha) \\ \hline\hline \mathcal{T}(L) \end{array}$$

$$= \begin{array}{c} \Gamma, \neg\alpha \vdash \alpha, \Delta \\ \hline\hline \Gamma, \neg\alpha \vdash \Delta \end{array} = \begin{array}{c} \Gamma, \neg\alpha \vdash \alpha, \Delta \\ \hline \Gamma, \neg\alpha, \neg\alpha \vdash \Delta \\ \hline \Gamma, \neg\alpha \vdash \Delta \end{array} \;\raise{1ex}{\neg\mathrm{R}}$$

- If $f = f_{F\neg}$:

$$\begin{array}{c} \mathcal{T}(f(F\alpha, L)) \\ \hline\hline \mathcal{T}(L) \end{array} = \begin{array}{c} \Gamma, \neg\alpha \vdash \neg\alpha, \Delta \\ \hline\hline \Gamma \vdash, \neg\alpha\Delta \end{array}$$

$$= \begin{array}{c} \Gamma, \alpha \vdash \neg\alpha, \Delta \\ \hline \Gamma \vdash \neg\alpha, \neg\alpha\Delta \\ \hline \Gamma, \vdash \neg\alpha\Delta \end{array} \;\raise{1ex}{\neg\mathrm{L}}$$

- If $f = f_{T\wedge}$:

$$\dfrac{\mathcal{T}(f(T(\alpha\wedge\beta),L))}{\mathcal{T}(L)} = \dfrac{\Gamma,\alpha,\beta,\alpha\wedge\beta\vdash\Delta}{\Gamma,\alpha\wedge\beta\vdash\Delta}$$

$$= \dfrac{\dfrac{\Gamma,\alpha,\beta\vdash\Delta}{\Gamma,\alpha,\beta,\alpha\wedge\beta\vdash\Delta}\;\wedge\mathrm{L}}{\Gamma,\alpha\wedge\beta\vdash\Delta}\;\text{contraction}$$

- If $f = f_{F\wedge}$:

$$\dfrac{\mathcal{T}(f(F(\alpha\wedge\beta),L))}{\mathcal{T}(L)} = \dfrac{\Gamma,\alpha,\alpha\wedge\beta\vdash\Delta \qquad \Gamma,\beta,\alpha\wedge\beta\vdash\Delta}{\Gamma,\vdash\alpha\wedge\beta,\Delta}$$

$$= \dfrac{\dfrac{\Gamma,\alpha,\alpha\wedge\beta\vdash\Delta \qquad \Gamma,\beta,\alpha\wedge\beta\vdash\Delta}{\Gamma\vdash\alpha\wedge\beta,\alpha\wedge\beta,\Delta}\;\wedge\mathrm{R}}{\Gamma,\vdash\alpha\wedge\beta,\Delta}\;\text{contraction}$$

- If $f = f_{T\vee}$:

$$\dfrac{\mathcal{T}(f(T(\alpha\vee\beta),L))}{\mathcal{T}(L)} = \dfrac{\Gamma,\alpha,\alpha\vee\beta\vdash\Delta \qquad \Gamma,\beta,\alpha\vee\beta\vdash\Delta}{\Gamma,\alpha\vee\beta\vdash\Delta}$$

$$= \dfrac{\dfrac{\Gamma,\alpha\vdash\alpha\vee\beta,\Delta \qquad \Gamma,\beta,\alpha\vee\beta\vdash\Delta}{\Gamma\vdash\alpha\vee\beta,\alpha\vee\beta,\Delta}\;\vee\mathrm{L}}{\Gamma\vdash\alpha\vee\beta,\Delta}\;\text{contraction}$$

- If $f = f_{F\vee}$:

$$\dfrac{\mathcal{T}(f(F(\alpha\vee\beta),L))}{\mathcal{T}(L)} = \dfrac{\Gamma\vdash\alpha\vee\beta,\alpha,\beta,\Delta}{\Gamma\vdash\alpha\vee\beta,\Delta}$$

$$= \dfrac{\dfrac{\dfrac{\dfrac{\Gamma\vdash\alpha\vee\beta,\alpha,\beta,\Delta}{\Gamma\vdash\alpha\vee\beta,\alpha\vee\beta,\beta\Delta}\;\vee R_1}{\Gamma\vdash\alpha\vee\beta,\alpha\vee\beta,\alpha\vee\beta\Delta}\;\vee R_2}{\Gamma\vdash\alpha\vee\beta,\alpha\vee\beta,\Delta}\;\text{contraction}}{\Gamma\vdash\alpha\vee\beta,\Delta}\;\text{contraction}$$

- If $f = f_{T\rightarrow}$:

$$\dfrac{\mathcal{T}(f(T(\alpha\rightarrow\beta),L))}{\mathcal{T}(L)} = \dfrac{\Gamma,\alpha,\alpha\rightarrow\beta\vdash\Delta \qquad \Gamma,\alpha\rightarrow\beta\vdash\beta,\Delta}{\Gamma,\alpha\rightarrow\beta\vdash\Delta}$$

$$= \dfrac{\dfrac{\Gamma,\alpha,\alpha\rightarrow\beta\vdash\Delta \qquad \Gamma,\alpha\rightarrow\beta\vdash\beta,\Delta}{\Gamma,\alpha\rightarrow\beta,\alpha\rightarrow\beta\vdash\Delta}\;\rightarrow\mathrm{L}}{\Gamma,\alpha\rightarrow\beta\vdash\Delta}\;\text{contraction}$$

- If $f = f_{F\to}$:

$$\begin{array}{c}\mathcal{T}(f(F(\alpha \to \beta), L)) \\ \hline\hline \mathcal{T}(L)\end{array} = \begin{array}{c}\Gamma, \alpha \vdash \beta, \alpha \to \beta, \Delta \\ \hline\hline \Gamma \vdash \alpha \to \beta, \Delta\end{array}$$

$$= \begin{array}{c}\Gamma, \alpha \vdash \beta, \alpha \to \beta, \Delta \\ \hline \Gamma \vdash \alpha \to \beta, \alpha \to \beta, \Delta \\ \hline \Gamma \vdash \alpha \to \beta, \Delta\end{array} \begin{array}{l}\to\text{R} \\ \text{contraction}\end{array}$$

- If $f = f_{T\forall}$:

$$\begin{array}{c}\mathcal{T}(f(T(\forall x\, \phi(x)), L)) \\ \hline\hline \mathcal{T}(L)\end{array} = \begin{array}{c}\Gamma, \forall x\, \phi(x), \phi(c_i) \vdash \Delta \\ \hline\hline \Gamma, \forall x\, \phi(x) \vdash \Delta\end{array}$$

$$= \begin{array}{c}\Gamma, \forall x\, \phi(x), \phi(c_i) \vdash \Delta \\ \hline \Gamma, \forall x\, \phi(x), \forall x\, \phi(x) \vdash \Delta \\ \hline \Gamma, \forall x\, \phi(x) \vdash \Delta\end{array} \begin{array}{l}\forall\text{L} \\ \text{contraction}\end{array}$$

- If $f = f_{F\forall}$:

$$\begin{array}{c}\mathcal{T}(f(F(\forall x\, \phi(x)), L)) \\ \hline\hline \mathcal{T}(L)\end{array} = \begin{array}{c}\Gamma \vdash \phi(c_i), \forall x\, \phi(x), \Delta \\ \hline\hline \Gamma \vdash \forall x\, \phi(x), \Delta\end{array}$$

$$= \begin{array}{c}\Gamma \vdash \phi(c_i), \forall x\, \phi(x), \Delta \\ \hline \Gamma \vdash \forall x\, \phi(x), \forall x\, \phi(x), \Delta \\ \hline \Gamma \vdash \forall x\, \phi(x), \Delta\end{array} \begin{array}{l}\forall\text{R} \\ \text{contraction}\end{array} \quad c_i \text{ not in } \Gamma \text{ or } \Delta\text{, as it is not in } L - \sigma\text{, fresh}$$

- If $f = f_{T\exists}$:

$$\begin{array}{c}\mathcal{T}(f(T(\exists x\, \phi(x)), L)) \\ \hline\hline \mathcal{T}(L)\end{array} = \begin{array}{c}\Gamma, \exists x\, \phi(x), \phi(c_i) \vdash \Delta \\ \hline\hline \Gamma, \exists x\, \phi(x) \vdash \Delta\end{array}$$

$$= \begin{array}{c}\Gamma, \exists x\, \phi(x), \phi(c_i) \vdash \Delta \\ \hline \Gamma, \exists x\, \phi(x), \exists x\, \phi(x) \vdash \Delta \\ \hline \Gamma, \exists x\, \phi(x) \vdash \Delta\end{array} \begin{array}{l}\exists\text{L} \\ \text{contraction}\end{array}$$

  c_i not in Γ or Δ, as it is not in $L - \sigma$, fresh

- If $f = f_{F\exists}$:

$$\begin{array}{c}\mathcal{T}(f(F(\exists x\, \phi(x)), L)) \\ \hline\hline \mathcal{T}(L)\end{array} = \begin{array}{c}\Gamma \vdash \phi(c_i), \exists x\, \phi(x), \Delta \\ \hline\hline \Gamma \vdash \exists x\, \phi(x), \Delta\end{array}$$

$$= \begin{array}{c}\Gamma \vdash \phi(c_i), \exists x\, \phi(x), \Delta \\ \hline \Gamma \vdash \exists x\, \phi(x), \exists x\, \phi(x), \Delta \\ \hline \Gamma \vdash \exists x\, \phi(x), \Delta\end{array} \begin{array}{l}\exists\text{R} \\ \text{contraction}\end{array}$$

Definition 17. ***Tree Translating Function*** *Given a tableaux proof-tree development τ and its root r , we define $\mathcal{T}_p(\tau)$:*

- *If $\tau = r$ and there are $T\sigma$ and $F\sigma$ on r:*
  $\mathcal{T}_p(\tau) = \dfrac{}{\mathcal{T}(r)}{}^{\,Ax\ \sigma}$
- *If $\tau = r$ and there is no σ such that $T\sigma$ and $F\sigma$ are on r:*
  $\mathcal{T}_p(\tau) = \mathcal{T}(\tau)$
- *If r has a single child r_0 with a corresponding subtree τ_0:*
  by definition, it exists a sentence σ such that $f(\sigma, r) = f_{Rule}(\sigma, r) = r_0$, and r_0 is the root of $\mathcal{T}_0$, so
  $\mathcal{T}_p(\tau) = \begin{array}{c} \mathcal{T}_p(\tau_0) \\ \hline\hline \mathcal{T}(r) \end{array}\ {}_{Rule\ on\ \sigma}.$
- *If r has two children r_1 and r_2 with corresponding subtrees τ_1 and τ_2:*
  by definition there exists a sentence σ such that $f(\sigma, r) = f_{Rule}(\sigma, r) = \{r_1, r_2\}$, and so:
  $\mathcal{T}_p(\tau) = \begin{array}{c} \mathcal{T}_p(\tau_1) \qquad \mathcal{T}_p(\tau_2) \\ \hline\hline \mathcal{T}(r) \end{array}\ {}_{Rule\ on\ \sigma}$

Theorem 5. *Given a classical tableaux development τ, $\mathcal{T}_p(\tau)$ is a valid sequent proof-tree development.*

Proof. Here r is the root of τ. The proof goes by induction on the size of τ:

- if $\tau =$ r and there are $T\sigma$ and $F\sigma$ on r, $\mathcal{T}_p(\tau) = \dfrac{}{\mathcal{T}(r)}{}^{\,\mathrm{Ax}\ \sigma}$ is a valid sequent proof.
- if $\tau =$ r and there are no $T\sigma$ and $F\sigma$ on r, $\mathcal{T}_p(\tau) =\ \mathcal{T}(r)$ is a valid sequent proof.
- if r has a single child r_0 with a corresponding subtree τ_0: $\mathcal{T}_p(\tau) = \begin{array}{c} \mathcal{T}_p(\tau_0) \\ \hline\hline \mathcal{T}(r) \end{array}\ {}_{\text{Rule on } \sigma}$ is a valid sequent proof since $\mathcal{T}_p(\tau_0)$ is valid by induction hypothesis and the rule is admissible by the Theroem 4.
- if r has two children r_1 and r_2 with corresponding subtrees τ_1 and τ_2: $\mathcal{T}_p(\tau) = \begin{array}{c} \mathcal{T}_p(\tau_1) \qquad \mathcal{T}_p(\tau_2) \\ \hline\hline \mathcal{T}(r) \end{array}\ {}_{\text{Rule on } \sigma}$ is a valid sequent proof since $\mathcal{T}_p(\tau_1)$ and $\mathcal{T}_p(\tau_2)$ are valid by induction hypothesis and the rule is admissible by the Theroem 4. □

Below there are some examples of tableaux proof trees and sequent proof trees obtained by the implementation of $\hookleftarrow_c$ and $\mathcal{T}_p$. A sequent proof verifier was also develped and used on the examples.

$$\begin{array}{c}
\mathrm{F}(\neg((a \to b) \land (a \land \neg b))) \\ \hline
\mathrm{T}(((a \to b) \land (a \land \neg b))),\ \mathrm{F}(\neg((a \to b) \land (a \land \neg b))) \\ \hline
\mathrm{T}((a \to b)),\ \mathrm{T}((a \land \neg b)),\ \ldots\ ,\ \mathrm{F}(\neg((a \to b) \land (a \land \neg b))),\ \mathrm{T}(((a \to b) \land (a \land \neg b))) \\ \hline
\begin{array}[t]{c}
\mathrm{T}((a \land \neg b)),\ \mathrm{F}(\neg((a \to b) \land (a \land \neg b))),\ \ldots\ ,\ \mathrm{T}((a \to b)),\ \mathrm{F}(a) \\ \hline
\mathrm{T}(a),\ \mathrm{T}(\neg b),\ \ldots\ ,\ \mathrm{F}(a),\ \mathrm{T}((a \land \neg b))
\end{array}
\quad
\begin{array}[t]{c}
\mathrm{T}((a \land \neg b)),\ \mathrm{F}(\neg((a \to b) \land (a \land \neg b))),\ \ldots\ ,\ \mathrm{T}((a \to b)),\ \mathrm{T}(b) \\ \hline
\mathrm{T}(\neg b),\ \mathrm{F}(\neg((a \to b) \land (a \land \neg b))),\ \ldots\ ,\ \mathrm{T}((a \land \neg b)),\ \mathrm{T}(a) \\ \hline
\mathrm{F}(b),\ \mathrm{F}(\neg((a \to b) \land (a \land \neg b))),\ \ldots\ ,\ \mathrm{T}(a),\ \mathrm{T}(\neg b)
\end{array}
\end{array}$$

Fig. 4. Example of an automatically generated tableaux proof tree.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{a, \ldots, (a \land \neg b) \vdash \neg((a \to b) \land (a \land \neg b)), a}\ \mathrm{Ax}\ a}{(a \land \neg b), ((a \to b) \land (a \land \neg b)), (a \to b) \vdash \neg((a \to b) \land (a \land \neg b)), a}\ \land\mathrm{L}\ (a \land \neg b) \qquad \dfrac{\dfrac{\dfrac{}{((a \to b) \land (a \land \neg b)), \ldots, \neg b \vdash b, \neg((a \to b) \land (a \land \neg b))}\ \mathrm{Ax}\ b}{\neg b, \ldots, a \vdash \neg((a \to b) \land (a \land \neg b))}\ \neg\mathrm{L}\ \neg b}{(a \land \neg b), \ldots, b \vdash \neg((a \to b) \land (a \land \neg b))}\ \land\mathrm{L}\ (a \land \neg b)}{(a \to b), (a \land \neg b), ((a \to b) \land (a \land \neg b)) \vdash \neg((a \to b) \land (a \land \neg b))}\ \to\mathrm{L}\ (a \to b)}{((a \to b) \land (a \land \neg b)) \vdash \neg((a \to b) \land (a \land \neg b))}\ \land\mathrm{L}\ ((a \to b) \land (a \land \neg b))}{\vdash \neg((a \to b) \land (a \land \neg b))}\ \neg\mathrm{R}\ \neg((a \to b) \land (a \land \neg b))$$

Fig. 5. Automatically generated translation of the tableaux proof tree of figure 4.

$$\begin{array}{c}
\mathrm{F}(\neg(x \land \neg\neg\neg x)) \\ \hline
\mathrm{T}((x \land \neg\neg\neg x)), \mathrm{F}(\neg(x \land \neg\neg\neg x)) \\ \hline
\mathrm{T}(\neg\neg\neg x), \mathrm{F}(\neg(x \land \neg\neg\neg x)), \ldots , \mathrm{T}((x \land \neg\neg\neg x)), \mathrm{T}(x) \\ \hline
\mathrm{F}(\neg\neg x), \mathrm{F}(\neg(x \land \neg\neg\neg x)), \ldots , \mathrm{T}(x), \mathrm{T}(\neg\neg\neg x) \\ \hline
\mathrm{T}(\neg x), \mathrm{F}(\neg(x \land \neg\neg\neg x)), \ldots , \mathrm{T}(\neg\neg\neg x), \mathrm{F}(\neg\neg x) \\ \hline
\mathrm{F}(x), \mathrm{F}(\neg(x \land \neg\neg\neg x)), \ldots , \mathrm{F}(\neg\neg x), \mathrm{T}(\neg x)
\end{array}$$

Fig. 6. Another example of an automatically generated tableaux proof tree.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{(x \land \neg\neg\neg x), \ldots, \neg x \vdash x, \neg(x \land \neg\neg\neg x), \neg\neg x}\ \mathrm{Ax}\ x}{\neg x, \ldots, \neg\neg\neg x \vdash \neg(x \land \neg\neg\neg x), \neg\neg x}\ \neg\mathrm{L}\ \neg x}{(x \land \neg\neg\neg x), x, \neg\neg\neg x \vdash \neg\neg x, \neg(x \land \neg\neg\neg x)}\ \neg\mathrm{R}\ \neg\neg x}{\neg\neg\neg x, (x \land \neg\neg\neg x), x \vdash \neg(x \land \neg\neg\neg x)}\ \neg\mathrm{L}\ \neg\neg\neg x}{(x \land \neg\neg\neg x) \vdash \neg(x \land \neg\neg\neg x)}\ \land\mathrm{L}\ (x \land \neg\neg\neg x)}{\vdash \neg(x \land \neg\neg\neg x)}\ \neg\mathrm{R}\ \neg(x \land \neg\neg\neg x)$$

Fig. 7. Automatically generated translation the tableaux proof tree of figure 6.

5.2 Intuitionistic Translation

When examining the intuitionistic tableaux proof of $\neg(A \wedge \neg A) \wedge (\neg X \wedge \neg Y)$ and attempting to choose appropriate words w for translation, we encounter an issue with the $\neg R$ rule: "useless" terms from our cumulative tableaux method can "overcrowd" the right side of the sequent equation. A first naive approach would be to remove all useless sentences. Another way to view this is that the sequent calculus has a Weakening rule that we have not yet utilized. We will show later on that this suficient for non-branching tableau.

Definition 18. ***Tree Thinning Function*** *Given an intuitionistic tableaux proof-tree development and its root r , we define $\mathcal{F}(\tau)$ ($\mathcal{F}$: tree $\rightarrow$ tree):*

- *If τ = r then there are $T_f\sigma$ and $F_f\sigma$ on r:*
  $\mathcal{F}(\tau) = [T_f\sigma, F_f\sigma]$
- *If r has a single child $f(\sigma, r)$*
  with a corresponding sub-tree τ_0 and r'_0 is the root of $\mathcal{F}(\tau_0)$:
  $(f(\sigma, r) - r)$ are the elements generated by the inference f and
  r'_0 are the elements necessary on the thinned proof $\mathcal{F}(\tau_0)$.
  In this case:
  - *If $(f(\sigma, r) - r) \cap r'_0 \neq \emptyset$*
    $\mathcal{F}(\tau)$ = a tree with root $(r'_0 - f(\sigma, [\sigma])) : \sigma$ connected to the child r'_0 : $(f(\sigma, [\sigma]) - \sigma)$, that has as a child the sub-tree $\mathcal{F}(\tau_0)$
  - *If $(f(\sigma, r) - r) \cap r'_0 = \emptyset$*
    $\mathcal{F}(\tau) = \mathcal{F}(\tau_0)$
- *If r has the children $f(\sigma, r)[1]$ and $f(\sigma, r)[2]$*
  with the corresponding sub-trees τ_1 and τ_2, r'_1 is the root of $\mathcal{F}(\tau_1)$ and r'_2 is the root of $\mathcal{F}(\tau_2)$, then $((f(\sigma, r)[1] - r) \cap r'_1) \cup ((f(\sigma, r)[2] - r) \cap r'_2)$ also defines if f was "useful":
  - *If $((f(\sigma, r)[1] - r) \cap r'_1) = \emptyset$: $\mathcal{F}(\tau) = \mathcal{F}(\tau_1)$*
  - *If $((f(\sigma, r)[2] - r) \cap r'_2) = \emptyset$: $\mathcal{F}(\tau) = \mathcal{F}(\tau_2)$*
  - *Otherwise: $\mathcal{F}(\tau)$ = a tree with root $(r'_1 - f(\sigma, [\sigma])[1]) : (r'_2 - f(\sigma, [\sigma])[2]) : \sigma$ connected to the children $r'_1 : (f(\sigma, [\sigma])[1] - \sigma)$ and $r'_2 : (f(\sigma, [\sigma])[2] - \sigma)$, each having $\mathcal{F}(\tau_1)$ and $\mathcal{F}(\tau_2)$, respectively, below them.*

We must, of course, be sure we have not removed "too much":

Theorem 6. *Given a tableaux proof τ, the root r' of $\mathcal{F}(\tau)$ is intuitionistically invalid*

Proof. The proof goes by induction on the size of τ:

- If τ = r, $\mathcal{F}(\tau) = [T_f\sigma, F_f\sigma]$. The theorem holds
- If r has a single child $f(\sigma, r)$
  with a corresponding sub-tree τ_0 and r'_0 is the root of $\mathcal{F}(\tau_0)$:
  - If $(f(\sigma, r) - r) \cap r'_0 \neq \emptyset$: $\mathcal{F}(\tau)$ has the root $(r'_0 - f(\sigma, [\sigma])) : \sigma$
    By induction hypothesis, we know r'_0 to be intuitionistically invalid. We also know that the intuitionistic validity of σ implies the intuitionistic validity of $f(\sigma, [\sigma])$:
    Assume r' to be valid, then $f(\sigma, r')$ should be valid then r'_0 should be also valid, (as $r'_0 \supseteq f(\sigma, r')$).This contradicts the induction hypothesis.

  - If $(f(\sigma, r) - r) \cap r'_0 = \emptyset$
    $\mathcal{F}(\tau) = \mathcal{F}(\tau_0)$
    And, by induction hypothesis, we know r'_0 to be intuitionistically invalid
- if r has the children $f(\sigma, r)[1]$ and $f(\sigma, r)[2]$:
  - If $(f(\sigma, r) - r) \cap (r'_1 \cup r'_2) \neq \emptyset$:
    $\mathcal{F}(\tau)$ has the root $(r'_0 - f(\sigma, [\sigma])) : \sigma$, Assume r' to be valid, then either $f(\sigma, r')[1]$ or $f(\sigma, r')[2]$ should be intuitionistically valid then either $r'_1 \supseteq f(\sigma, r')[2]$ or $r'_1 \supseteq f(\sigma, r')[2]$ should be intuitionistically valid. We know, by induction hypothesis, that both are intuitionistically invalid, which yields a contradiction.
  - If $(f(\sigma, r) - r) \cap (\mathcal{F}(r'_1) \cup \mathcal{F}(r'_2)) = \emptyset$:
    $\mathcal{F}(\tau) = \mathcal{F}(\tau_1)$ is valid by induction hypothesis. □

Theorem 7. *If a tableaux proof-tree τ does not branch, the root r' of $\mathcal{F}(\tau)$ has not more than 2 signed sentences*

Proof. The proof goes by induction on the size of τ:

- If τ = r, $\mathcal{F}(\tau) = [T_f\sigma, F_f\sigma]$. The theorem holds
- If r has a single child $f(\sigma, r)$:
  with a corresponding sub-tree τ_0 and r'_0 is the root of $\mathcal{F}(\tau_0)$:
  - If $(f(\sigma, r) - r) \cap r'_0 \neq \emptyset$:
    the root r' of $\mathcal{F}(\tau)$ is $(r'_0 - f(\sigma, [\sigma])) : \sigma$
    by induction hypothesis $\#(r'_0) \leq 2$ and so $\#[(r'_0 - f(\sigma, [\sigma])) : \sigma] \leq 2$
  - If $(f(\sigma, r) - r) \cap r'_0 = \emptyset$
    $\mathcal{F}(\tau) = \mathcal{F}(\tau_0)$
    and so $\#(r') = \#(r'_0) \leq 2$ □

A first goal would be to prove that the non-branching parts (the last branches) of a thinned tableaux never have two signed sentences of type F. One way would be to use the fact that when assuming A and B not valid, a sequent of type $\vdash A, B$ is not provable in intuitionistic sequent calculus. By the completeness of intuitionistic sequent calculus that would mean that the sequent $\vdash A, B$ is invalid. A proof in this direction would not be interesting, as it would use the translation to demonstrate a property of the translation that we are trying to show.

A less semantics-oriented approach would be to look at the general format that the root r' of a thinned not branching tableau $\mathcal{F}(t)$ can have. Knowing that a contradiction always has the format $F_p()T_q()$, with $p = q$, we can inspect the formats of the sequence lists that could have generated it.

Figure 8 can be an example for the representation: an arrow goes from $X_p()Y_q()$, with prq to $X'_p()Y'_q()$ with $pr'q$ if the root of a $\mathcal{F}(\tau)$ can have the format $X_p()Y_q()$ with prq and the root of $\mathcal{F}(\hookleftarrow (F\neg\alpha, \tau))$ can have the format $X'_p()Y'_q()$ with $pr'q$.

If we were to do all of them, we can inspect that the root of a thinned not branching tableaux proof can never have the formats
$F_p()F_q()$ with $p = q$,
$F_p()F_q()$ with $p > q$,
$T_p()F_q()$ with $p > q$. This exaustion, shown in figure 10, proves that these states never reach the state [$F_p()T_q()$, with $p = q$s]

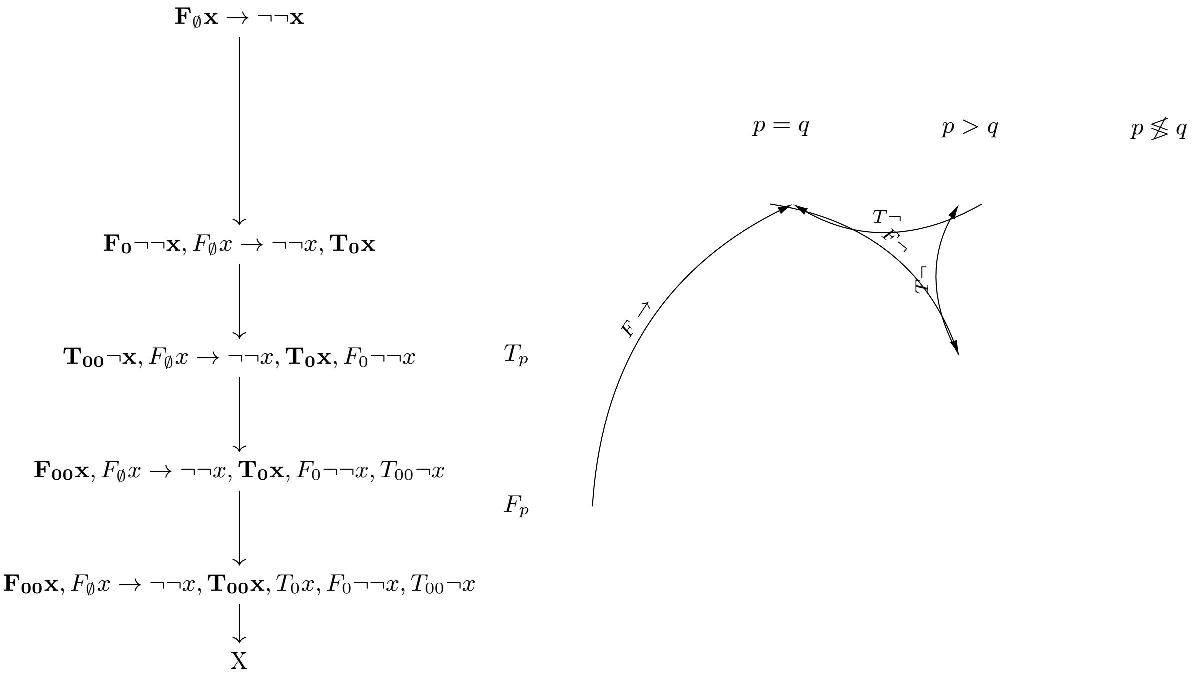
$\mathbf{F_\emptyset x} \to \neg\neg\mathbf{x}$
$\mathbf{F_0}\neg\neg\mathbf{x}, F_\emptyset x \to \neg\neg x, \mathbf{T_0 x}$
$\mathbf{T_{00}}\neg\mathbf{x}, F_\emptyset x \to \neg\neg x, \mathbf{T_0 x}, F_0\neg\neg x$
$\mathbf{F_{00} x}, F_\emptyset x \to \neg\neg x, \mathbf{T_0 x}, F_0\neg\neg x, T_{00}\neg x$
$\mathbf{F_{00} x}, F_\emptyset x \to \neg\neg x, \mathbf{T_{00} x}, T_0 x, F_0\neg\neg x, T_{00}\neg x$
X
$p = q$
$p > q$
$p \ngtrless q$
T_p
F_p
$F_\to$
$T_\neg$
$F_\neg$
$F_\neg$

Fig. 8. Example of a thinned tableaux proof tree and its transitions.

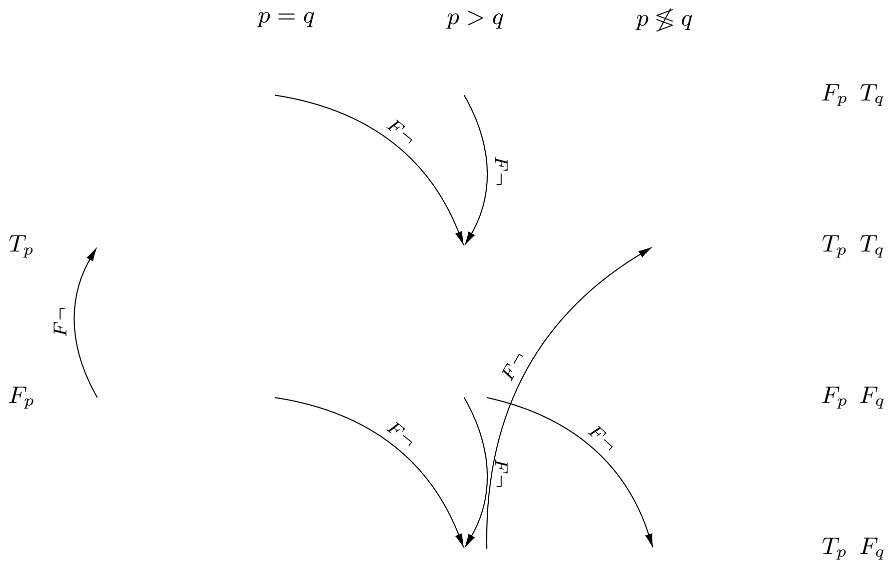

Fig. 9. The possible transitions between formats of nodes in a thinned tableau, possible by the $F_\neg$ rule. Translations in incomparable formats were omited, as they never lead to a contradiction.

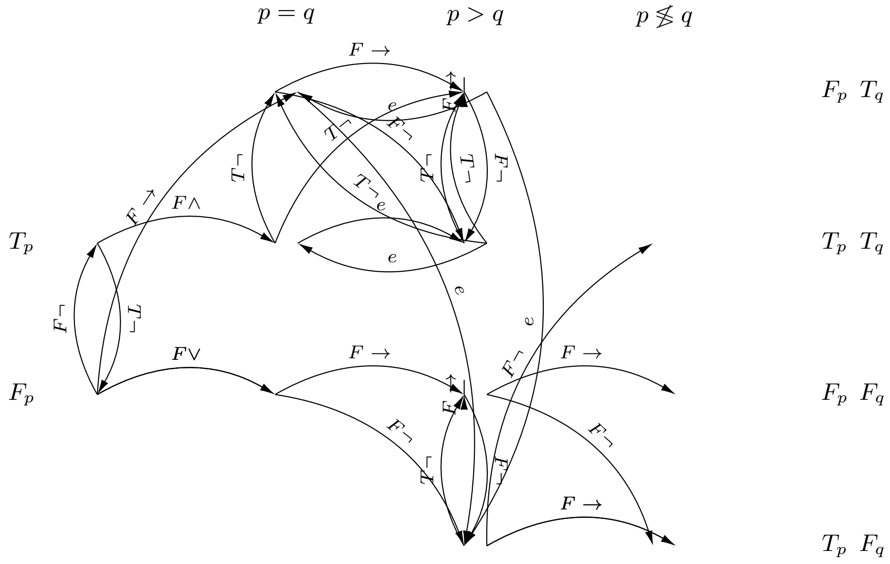

Fig. 10. The possible transitions between formats of nodes in a thinned tableaux. Translations in incomparable formats were omited, as they never lead to a contradiction.

6 Implementation

The work was implemented in OCAML, and the code is available at `github.com/lontra-lontra/proof-translations`. The implementation of the classical tableaux method and the translation to sequent calculus is complete, and the intuitionistic logic case is partially implemented. More precisely:

- The classical tableaux method, classical sequent calculus, classical transilation and classical sequent calculus proof verification are implemented in the folder **classical**.
- The intuitionistic definitions from 1.2 are implemented in the folder **intuitionistic_defintions**.
- The intuitionistic tableaux method is partially implemented in the folder **intuitionistic_tableaux**.
- The main function is in the main file, it contains some possible configurations for testing the implementation.
- The output folder contains some .tex files with the representation of some of the structures seen in this document, as well as some automatically generated tableaux proofs and their translations to sequent calculus.

The more up-to-date correspandnce between the OCAML code and the definitions in this document can be found in the README.md file in the root of the repository.

7 Conclusion

This work was an exercise in several important aspects of research — notably, the deep comprehension and clear expression of the subjects studied. The github repository presents the the OCAML generation of classical tableaux proofs, translation to sequento sequent calculus proofs, and the verification of the classical sequent-calculs proofs. Some progress was made in the intuitionistic logic case. The next step would be to analyze the translatability behavior in branching nodes in the intuitionistic logic case.

References

1. Nerode, A., Shore, R.A.: Logic for Applications. 2nd edn. Texts in Computer Science. Springer, New York, NY (1997). https://doi.org/10.1007/978-1-4612-0649-1
2. Dummett, M.: Elements of Intuitionism. Oxford Logic Guides. Clarendon Press (2000). https://books.google.fr/books?id=JVFzknbGBVAC
3. Kripke, S.A.: Semantical analysis of intuitionistic logic I. In: Studies in Logic and the Foundations of Mathematics. Elsevier (1965)
4. Svejdar, V.: On Sequent Calculi for Intuitionistic Propositional Logic. CMUC (2005). https://www.muni.cz/en/research/publications/1234567