Universiteit Utrecht

Faculteit Bètawetenschappen

Proof Systems and the Correspondence between Cut Elimination in Sequent Calculus and Normal Forms in Natural Deduction

Bachelor Thesis

Joost van der Laan

Wiskunde

Supervisor:

Dr.J. van Oosten Supervisor

June 2023

Abstract

Normalisation and cut elimination are methods used to perform combinatoral analysis on the structure of formal proofs, while translations between proof systems are used to analyse and interpret the meaning of proofs. In this thesis I will introduce two systems and their respective normal forms and demonstrate how to perform translations between the systems which will preserve these normal forms.
This thesis also serves as an introduction to proof theory, In the first chapters the systems of natural deduction and sequent calculus and their normal forms will be introduced from scratch. I will also demonstrate an error with the cut elimination proof in [1] and give a correction, this serves as an alternative to the correction given in [2].

Contents

1 Proof systems

For most subjects I cover: My definitions will be slightly different to help with later proofs, make equivalences more explicit, or just to investigate these systems in a different manner. Most of the theory in this thesis is from 'Natural Deduction' by Prawits [3] and 'An Introduction to proof-theory' by Buss [1].

In chapter 1 I will give an introduction to proof theory. Defining statements, models and proof trees. In Chapter 2 I introduce Hilbert-Style proof systems, which serve as an introduction as they are quite simple systems, next I will define Natural Deduction and finally I will show that Natural Deduction and Hilbert Style Systems are equivalent. I have constructed an alternative proof myself to show that Hilbert Style Systems model Natural Deduction.

In chapter 3 I will define the sequent calculus and prove completeness. In this thesis we will give the reader an introduction to several different proof-systems. We will then compare these systems and investigate their equivalence.
In chapter 4 I will demonstrate a proof of cut elimination as shown in [1], I will then demonstrate an error with this proof and correct it.

In chapter 5 I will introduce normal forms on Natural deduction and properties of those normal forms, these normal forms will allow us to prove equivalence between natural deduction and sequent calculus.
In chapter 6 I will define a normal form on sequent calculus proof trees, I will then show that these normal forms are equivalent. The converse of this proof is my own. After this proof we will investigate further into the correspondence between these normal forms.

1.1 Statements

Proof systems give us methods to prove statements. Instead of proving statements written in English or any other spoken language, we define our own languages. This is necessary in order to remove ambiguity and paradoxes. These definitions have been taken from [3], with slight modifications.

Definition 1.1.1 (Language)**.** A language contains symbols with which we can construct our statements. A language L by definition contains the following

- Free variables: $a, b, c, ...$
- Bound variables $..., x, y, z.$
- Binary connectives $\supset$, $\vee$ and $\wedge$.
- Relational constant $\bot$.
- Quantifier symbols $(\forall)$, $(\exists)$, all quantifier symbols are unary connectives.

For each kind of variable L contains an infinite number. We will use subscript $c_1, c_2, c_3, ...$ if necessary.
We can have different languages to express different kinds of statements. To facilitate this expression a language may contain any number of the following:

- Constants: $a, b, c, ...$
- For every integer k, function symbols $f, g, h, ...$ with arity k.
- For every integer k, relational symbols $R, U, V, ...$ with arity k.

Remark 1.1.2. We have defined three different kinds of variables. Their exact use will become clear later, but we will now give some intuition each category.
Constants are variables that have an interpretation, they are often used in axiom schemes. The numbers 0 and 1 are often used as constants.
Free variables do not have an interpretation, they could be any constant. It is similar to if we use a number

n in definitions. In this, n could be any constant.
Bound variables are similar to free variables, but play a very specific role. Bound variables occur in statements like "For all x we have $f(x) = 0$". They are used as a proxy to write in 'for all' or 'there exists' statements.

Definition 1.1.3 (Terms)**.** We inductively define terms $t, s, r, ...$ as follows

i) Every free variable is a term

ii) For every k-ary function f and terms $t_1, ..., t_k$, the string $f(t_1, ..., t_k)$ is a term.

We define terms containing any number of bound variables as *pseudo-terms*. Any term is also a pseudo-term.

Definition 1.1.4 (Substitution)**.** In a term s, we can substitute any string t. We write $s[t/x]$ for replacing every occurrence of x in s with the string t. If t is a term then the resulting string $s[t/x]$ will also be a term. For variables x, y with $x \neq y$, f a k-ary function and terms $t_1, ...t_k$ we define the following:

- $x[t/x] := t$
- $y[t/x] := y$
- $f(t_1, ..., t_k)[t/x] := f(t_1[t/x], ..., t_k[t/x])$

We write $[t_1/x_1, ..., t_n/x_n]$ for simultanious substitution of multiple terms.

Definition 1.1.5 (atomic formulas)**.**

- The relational constant $\perp$ is an atomic formula.
- For every k-ary relational symbol R and terms $t_1, ..., t_k$ we define $R(t_1, ..., t_k)$ as an atomic formula.

Definition 1.1.6 (formulas)**.** We inductively define formulas, firstly any atomic formula is a formula.
For any formulas A and B we define the following as formulas:

i) For formulas A and B the string $(A \supset B)$ is a formula.

ii) For formulas A and B the string $(A \wedge B)$ is a formula.

iii) For formulas A and B the string $(A \vee B)$ is a formula.

iv) For a formula A, variable a and bound variable x, the string $((\forall x)A[x/a])$ is a formula.

v) For a formula A, variable a and bound variable x, the string $((\exists x)A[x/a])$ is a formula.

In definitions (iv) and (v), we say x is bound by $(\forall x)$ or $(\exists x)$ respectively.

Definition 1.1.7 (Substitution in formulas)**.** Similarly to terms, we also allow for substitution in formulas: We write $A[t/x]$ for replacing all occurrences of variable x with the term t in A.
Specifically, for any relational symbol R, term t and variable x we define

- $R(t_1, ..., t_k)[t/x] := R(t_1[t/x], ..., t_k[t/x])$
- $\perp[t/x] := \perp$.

Furthermore, if $\odot$ is any binary connective i.e. $\odot \in \{\supset, \wedge, \vee\}$, we define

- $(A \odot B)[t/x] := ((A[t/x]) \odot (B[t/x]))$.

Bound variables may only be bound once, we will structure our definitions in order to enforce this.[1] Let Q a quantifier symbol, i.e. $\exists$ or $\forall$. Let x a bound variable, t a term and a a variable not equal to x. We now define:

- $((Qx)A)[t/x] := (Qx)A$

[1] These definitions are inspired by similar restrictions in λ-calculus [4].

- $((Qx)A)[t/a] := ((Qx)A[t/a])$ for x not in t.
- $((Qx)A)[t/a] := (Qy)(A[y/x][t/a])$ for x in t. Where we y is a new bound variable not in t.

Definition 1.1.8 (short-hand notation)**.**

- We use $\neg$ as a unary connective. We will write $\neg A$ for $A \supset \bot$.
- We use $\equiv$ as a binary connective. We will write $A \equiv B$ for $(A \supset B) \wedge (B \supset A)$
- We will omit outer parentheses.

When we omit any further parentheses, we define *precedence* to each of the connectives. Precedence gives a precise definition for the order of operations. Connectives with higher precedence will bind earlier and bind the closest formulas to them.

- Unary connectives have the highest precedence. For any unary connective $U \in \{(\forall x), (\exists x), \neg\}$, we may write $UA \wedge UB$ for $((UA) \wedge (UB))$.
- The binary connective $\supset$ has the lowest precedence:
  We may write $A \wedge B \supset B \vee A$ for $((A \wedge B) \supset (B \vee A))$,
  while $A \supset B \wedge B \supset A$ is short for $A \supset (B \wedge B) \supset A$.
- The connective $\vee$ has higher precedence than $\wedge$.
- For binary connectives with the same precedence we associate from right to left, i.e. the rightmost connective has the highest precedence.
  We may write $A \supset B \supset C$ for $A \supset (B \supset C)$

Definition 1.1.9 (Sub-formulas)**.** Because we defined formulas inductively, we can define an *outer connective* for every non-atomic formula C. If C is of the form $(Qx)A$ then the unary connective Q, i.e. $\forall$ or $\exists$. is the outer connective of C. Similarly if C is of the form $A \odot B$ for any binary connective $\odot \in \{\supset, \wedge, \vee\}$, then $\odot$ is the outer connective of C.
In both of these cases, we define A (and B) as *immediate-sub-formulas* of C. We also define *sub-formulas* as the reflexive and transitive closure of immediate-sub-formulas.

1.2 Models

In this section we will introduce *models*. Our definition of models has been taken from [1], but rewritten and slightly modified.

In the previous section we introduced statements in the form of formulas. Models are then used to give an interpretation of these statements and assign truth to them.

Definition 1.2.1 (Model)**.**
A model $\mathcal{M}$ of a language L contains the following:

- A universe of objects $\mathcal{M}$, with for every constant c a corresponding interpretation $c^M \in \mathcal{M}$.
- For every k-ary function f, an interpretation $f^M : M^k \to M$.
- For every k-ary relational symbol R, we have an interpretation $R^M \subseteq M^k$.containing all k-tuples defined as true under relation R.

Using these definitions we can find the interpretations of terms. We define $(f(t_1, ..., t_k))^M := f^M(t_1^M, ..., t_k^M)$ which will correspond to a single term in $\mathcal{M}$.

Remark 1.2.2. In this we interpret $(t_1^M, ..., t_k^M) \in R^M$ as $R\, t_1, ..., t_k$ being true in $\mathcal{M}$. Our model may admit more objects besides the ones defined above. This could influence $\exists$ and $\forall$ statements and should be taken in account. Note that only constants have an interpretation, as bound and free variables may correspond to "any" interpretation.

Definition 1.2.3. We say $\mathcal{M}$ *models* an atomic formula $R(t_1, ..., t_k)$ if and only if $(t_1^M, ..., t_k^M) \in R^M$. We will denote this by $\mathcal{M} \models R(t_1, ..., t_k)$. Informally we can interpret $\mathcal{M} \models R(t_1, ..., t_k)$ as $R(t_1, ..., t_k)$ being true in $\mathcal{M}$.

For non-atomic formulas we inductively define the relation $\models$ as follows:

- We say $M \models A \wedge B$ if and only if $M \models A$ and $M \models B$.
- We say $M \models A \vee B$ if and only if either $M \models A$ or $M \models B$.
- We say $M \models A \supset B$ if and only if $M \not\models A$ or $M \models B$.
- We define $M \not\models \bot$,
- We have $M \models \forall x F$ if and only if for every $a \in M$ we have $M \models F[a/x]$.
- We have $M \models \exists x F$ if and only if there exists an $a \in M$ for which $M \models F[a/x]$.

Where A, B and F may be any formula. If $\mathcal{M}$ does not model a formula C we write $\mathcal{M} \not\models C$. We can consider this as C not being true in $\mathcal{M}$ and is equivalent to $\mathcal{M} \models \neg C$, which is an abbreviation of $\mathcal{M} \models C \supset \bot$.

Definition 1.2.4. Let Γ be a set of formulas and $\mathcal{M}$ a model. We say $\mathcal{M} \models \Gamma$ if and only if for every $B \in \Gamma$ we have $\mathcal{M} \models B$.

Definition 1.2.5. Let Γ a set of formulas and A a formula. We say Γ models a A if and only if for every model $\mathcal{M}$ with $\mathcal{M} \models \Gamma$ we have $\mathcal{M} \models A$. We denote this by writing $\Gamma \models A$.

We may write $\models A$ for Γ empty. This is equivalent to $\mathcal{M} \models A$ for every model $\mathcal{M}$.

Remark 1.2.6 (Evaluating Systems)**.** The definition above is what we will most often use to evaluate truth. The truth of a statement is often dependent on what axioms we are working under. the set Γ will then contain our axioms. A formula A will be true under those axioms (i.e. $\Gamma \models A$) if and only if for every model that satisfies those axioms: A is also true.

Remark 1.2.7 (Contradictions)**.** Our definition of truth for atomic formulas is very simple and leaves no room for contradictions, each tuple is either in R^M or it is not. We then easily extend this definition to non-atomic formulas. Because models are quite simple, and because their validity is easily verified, they are a great tool to evaluate truth of statements. Models will serve as our sanity check for the validity of proof-systems.

1.3 Notation

In this section we will introduce the general structure of a proof-tree. We will focus on the common structure and terminology that will be present in most proof-systems.

We can represent every proof as a graph, specifically we will only consider finite tree-like proofs. Formulas are nodes, nodes with no incoming vertices are called *leaves* or *assumptions* of our proof. There is exactly one formula at the root of our tree, the only formula with no outgoing vertex. This formula will be named our *end-formula* or the conclusion of our proof.

Each different proof system has different rules for how to construct these trees. Generally each system has a set of assumption trees each consisting of one node we can start with. Furthermore each system will have a set of *inference rules*. Given a set of proof trees, an inference rule allows us to create a new node (formula) and connect the conclusions of our proof-trees to this new formula, effectively combining proof-trees in order to reach a new conclusion. These rules are dependent on the end-formula of the used proof-trees.

These inference rules can be written as $\Sigma(A_1, ..., A_n, B)$ i.e. we can take n proof-trees with conclusions $A_1, ..., A_n$ respectively and connect their end-formulas to a new node B. Informally this signifies we can conclude a formula B from assumptions $A_1, ..., A_n$.
An example of this would be $\Sigma(C, D, C \wedge D)$, i.e. given C and D we can conclude $C \wedge D$. Instead of writing these formulas we will often use the following notation.

$$\frac{C \qquad D}{C \wedge D}$$

Within this inference rule we call C and D *upper formula* and $C \wedge D$ the *lower formula*, we also use *assumption-* and *conclusion formula* respectively. We will often speak of *formula occurrences* in proof-trees, which signify a specific node. We say two formula occurrences of a formula A have the same *shape*. We write $[A]$ to signify the set of formulas of shape A.

If Γ is the set of leaves, and A the end formula, we write $\Gamma \vdash A$ signifying there exists a proof tree from Γ to A. The specific proof system will be clear from context or specified otherwise. We also say that A can be deduced from Γ in this system. Often when a proof system uses axioms, we will not include those in Γ.

Remark 1.3.1 (Properties of Proof Systems)**.** Some of the most important properties of proof systems are Soundness, Completeness and Consistency. These verify the validity of our proof system and we will prove most of them for each system we introduce.

Definition 1.3.2 (Soundness)**.** A proof system is sound if $\vdash A$ implies that $\models A$. A proof system is implicationally sound if $\Gamma \vdash A$ implies $\Gamma \vDash A$ for any Γ.

Definition 1.3.3 (Completeness)**.** A proof system is complete if $\vDash A$ implies $\vdash A$. Similarly a proof system is implicationally complete if $\Gamma \models A$ implies $\Gamma \vdash A$.

Definition 1.3.4 (Consistency)**.** A proof system is consistent if you cannot prove contradictory formula(s). Specifically this means we cannot prove $\vdash \bot$.

Remark 1.3.5. Consistency follows from soundness, as models cannot contain contradictions. This does not tell us much about the proof system in question however. More constructive proofs will tell us more about these systems. Being able to easily prove each of these properties shows we have a robust system.

Definition 1.3.6 (Equivalence)**.** We say a proof system K *models* a proof system L if every proof tree in L with end-formula A can be rewritten as a proof tree in K with the same end-formula.
Two proof systems are equivalent if each of them models the other.

2 Natural Deduction and Hilbert-style Systems

2.1 Hilbert-Style Proof systems

In this chapter we will introduce Hilbert-style systems and prove some of the properties introduced in 1.3. Hilbert-style systems are interesting from a historical perspective as they were one of the first tree-like proof systems. This system is simple and equivalent to Natural deduction, which has been proven by Gentzen [5]. For these reasons the Hilbert-style proof systems make for a fine introduction to proof systems.

We will define Hilbert-style systems and a unique axiom scheme. Using this new axiom scheme we will replicate proofs of soundness and completeness given in [1]. In 2.3 we will prove the equivalence between Hilbert-style systems and natural deduction.

Hilbert-style proof systems have only one inference rule: *modus ponens*. We can infer

$$\frac{A \qquad A \supset B}{B}$$

for any formulas A and B. This simplicity in our inference rules requires a rather large set of axioms. We will be using an axiom schema inspired by that of Frege. We will first introduce a set of structural axioms, whose names have been chosen to mirror the inference rules in sequent calculus:

$$A \supset B \supset A \tag{Constant}$$
$$(A \supset A \supset B) \supset (A \supset B) \tag{Contraction}$$
$$(A \supset B \supset C) \supset (B \supset A \supset C) \tag{Exchange}$$
$$(A \supset B) \supset (B \supset C) \supset (A \supset C). \tag{Transitivity}$$

Furthermore we have a set of propositional axioms, each labeled as a shorthand of the rule itself.

$$A \supset A \vee B \qquad (\to \vee)$$
$$B \supset A \vee B \qquad (\to \vee)$$
$$(A \supset C) \supset (B \supset C) \supset (A \vee B \supset C) \qquad (\vee \to)$$
$$A \wedge B \supset A \qquad (\wedge \to)$$
$$A \wedge B \supset B \qquad (\wedge \to)$$
$$A \supset B \supset A \wedge B \qquad (\to \wedge)$$
$$\neg\neg A \supset A \qquad (\bot \to)$$

.

For this thesis we will simply name this system $\mathcal{H}_p$. Assumption trees contain one of the formulas above, in this we allow for substitution. We may replace A, B and C for any formula of our choosing.

We can only construct further proof trees using the modus ponens inference rule. Using a proof tree with end-formula $A \supset B$ and one with end-formula A we can create a new proof tree by connecting thesse end formulae to our new conclusion B. We will continue this process until we reach our desired conclusion.

If we wanted to substitute a formula into the end-formula of a proof tree, we would need to propagate this change upwards to the leaves of our tree. This can create rather large formulas however, specifically in the axioms of our proof tree. Will sometimes informally use substitution when displaying proof trees, in order to make proof trees more readable.

Remark 2.1.1. We have chosen this set of axioms for a few reasons: As mentioned we have tried to draw out the equivalence to sequent calculus, this also shows the symmetry in the axioms themselves.
Specifically these axioms were chosen to make our equivalence proof in 2.3 both shorter and more clear.

Equivalence between different axiom schemes in Hilbert style systems is very easy to proof. Any of Frege's axioms can easily be inferred in $\mathcal{H}_p$, therefore any proof tree in $\mathcal{F}$ can be transformed to a proof in $\mathcal{H}_p$ by replacing each of these axioms with a $\mathcal{H}_p$ proof of this axiom.

Theorem 2.1.2 (Soundness[1])**.** *The propositional proof system $\mathcal{H}_p$ is implicationally sound, if $\Gamma \vdash C$ then $\Gamma \models C$.*

Proof. Let Σ a proof tree with end-formula C. Every leaf A in Σ is either contained in Γ or an axiom, the truth of axioms is very easy to verify. We can therefore conclude that $\Gamma \models A$. If for two formulas A, B we have $\Gamma \models A$ and $\Gamma \models A \supset B$, then per definition (1.2.3) we find B must hold in all models that satisfy Γ. Therefore using induction on the size of our proof trees we find that indeed $\Gamma \models C$. □

Remark 2.1.3. We will illustrate a proof of completeness as given in [1]. For this we will first need to introduce a few lemmas. Lemma (2.1.6) shows that if we know the truth of all atomic sub-formulas in a formula A then from those sub-formulas follows either A or $\neg A$.

In lemma (2.1.5) We show that if B follows from A but also follows from $\neg A$ then B is true without needing the axiom A.

Lemma 2.1.4. *If* $\Gamma, A \vdash B$, *then* $\Gamma \vdash A \supset B$.

Proof. Let Σ a proof of B from Γ, A. Using induction on the size of Σ we will show we can transform this to a proof Σ' of $A \supset B$ from Γ. For our base case we will assume Σ is an assumption tree containing only the formula D. If $D = A$ we use the constant axiom $A \supset A \supset A$ and a contraction to infer $A \supset D$.
If $D \neq A$ then using the constant axiom $D \supset A \supset D$, we create the following proof tree Σ

$$\frac{D \qquad D \supset A \supset D}{A \supset D}$$

Let Σ a proof tree with at least one inference, this will look as follows

$$\frac{\begin{matrix}\Sigma_1 \\ C\end{matrix} \qquad \begin{matrix}\Sigma_2 \\ C \supset D\end{matrix}}{D}$$

Per induction assumption we can apply our transformation to both Σ_1 and Σ_2 to create proof trees Σ_1' and Σ_2' with end-formulas $A \supset C \supset D$ and $A \supset C$ respectively. We can now construct Σ' as follows

$$\dfrac{\dfrac{\begin{matrix}\Sigma_2' \\ A \supset C \supset D\end{matrix} \qquad (A \supset C \supset D) \supset (C \supset A \supset D)}{C \supset A \supset D} \qquad \dfrac{\begin{matrix}\Sigma_1' \\ A \supset C\end{matrix} \qquad (X \supset Y) \supset (Y \supset Z) \supset (X \supset Z)}{(C \supset (A \supset D)) \supset (A \supset (A \supset D))}}{A \supset A \supset D}$$

Applying the contraction axiom to the end-formula gives us our desired result. Per induction we can now transform any proof tree Σ from Γ, A to B into a proof tree Σ' of $\Gamma \vdash A \supset B$ □

Lemma 2.1.5. *If* $\Gamma, A \vdash B$ *and* $\Gamma, \neg A \vdash B$, *then* $\Gamma \vdash B$.

Proof. From (2.1.4) we find $\Gamma \vdash A \supset B$ and $\Gamma \vdash \neg A \supset B$.
We can use the transitivity axiom as follows:

$$\frac{\begin{matrix}\Sigma_1 \\ A \supset B\end{matrix} \qquad (A \supset B) \supset (B \supset \bot) \supset (A \supset \bot)}{\neg B \supset \neg A}$$

- Applying the same transformation to $\neg A \supset B$, we find $\neg B \supset \neg\neg A$. This formula is shorthand for $\neg B \supset (\neg A) \supset \bot$ and applying the exchange axiom gives us the result $\neg A \supset \neg\neg B$.
- Applying the transitivity axiom on $\neg B \supset \neg A$ and $\neg A \supset \neg\neg B$ we find $\neg B \supset \neg\neg B$. We can rewrite this as $\neg B \supset (\neg B) \supset \bot$, using contraction we may conclude $\neg\neg B$
- Finally applying $(\bot \rightarrow)$ i.e. $\neg\neg B \supset B$ gives us our desired result B.

□

Lemma 2.1.6. *For any formula A which contains no quantifier symbols. We define $B_1, ..., B_k$ as the sequence of all atomic sub-formulas in A. For any sequence $\Gamma := C_1, ..., C_k$ such that C_i is either B_i or $\neg B_i$, we have either*

$$C_1, ..., C_k \vdash A \text{ or } C_1, ..., C_k \vdash \neg A$$

We prove this using induction on the complexity of A.
Case 1): If A is atomic the resulting proof tree is trivial. Either for for A is B_i we need to prove $C_i \vdash C_i$, or for A is $\bot$ we prove $\Gamma \vdash \bot \supset \bot$.
For the induction cases, our induction assumption will be that for every subformula D of A we have either $\Gamma \vdash D$ or $\Gamma \vdash \neg D$.

Case 2): A has shape $D \vee E$, then if either $\Gamma \vdash D$ or $\Gamma \vdash E$ we can use $(\to \vee)$ to infer $\Gamma \vdash A$. If on the other hand $\Gamma \vdash \neg D$ and $\Gamma \vdash \neg E$. Then with axiom $(\vee \to)$ we can use $(D \supset \bot) \supset (E \supset \bot) \supset (D \vee E \supset \bot)$ and conclude $\Gamma \vdash \neg A$.

Case 3): A has shape $D \wedge E$. If both $\Gamma \vdash D$ and $\Gamma \vdash E$ then $(\to \wedge)$:$D \supset E \supset D \wedge E$ gives us the desired result. If on the other hand either formula is false: without loss of generality assume $\Gamma \vdash D \supset \bot$, then using transitivity we get $(D \wedge E \supset D) \supset (D \supset \bot) \supset (D \wedge E \supset \bot)$, where $(D \wedge C) \supset D$ is the $(\wedge \to)$ axiom.

Case 4): A has shape $D \supset E$. We seperate three possible subcases:

- *4a): $\Gamma \vdash E$, we can now use the* (Constant) *axiom $E \supset D \supset E$ and modus ponens to conclude $D \supset E$.*
- *4b): $\Gamma \vdash \neg D$, we first note that again using the constant axiom we can find $\bot \supset \neg\neg E$, then using transitivity and $(\bot \to)$ we find $\bot \supset E$. Finally if we use transitivity again we find $(D \supset \bot) \supset (\bot \supset E) \supset (D \supset E)$, from which we obtain $\Gamma \vdash D \supset E$.*
- *4c): $\Gamma \vdash D$ and $\Gamma \vdash E \supset \bot$. Transitivity gives us $(D \supset E) \supset (E \supset \bot) \supset (D \supset \bot)$. Using the exchange axiom and $E \supset \bot$ we can simplify to $(D \supset E) \supset (D \supset \bot)$ if we again use the exchange axiom to receive $D \supset (D \supset E) \supset \bot$. Finally using modus ponens with D we conclude $(D \supset E) \supset \bot$. Our desired result.*

Theorem 2.1.7 (Completeness)**.** *The propositional proof system $\mathcal{H}$ is complete, namely for any formula A containing no quantifier symbols: if $\models A$ then $\vdash A$.*

Proof. Assume $\models A$, let $B_1, ..., B_n$ be all atomic formulas in A. For every B_i we define $C_i^0 = B_i$ and $C_i^1 = \neg B_i$. For any $(p_1, ..., p_n) \in \{0,1\}^n$ we can easily construct a model $\mathcal{M}$ such that $\mathcal{M} \models C_1^{p_1}, ..., C_n^{p_n}$. In particular because $\models A$, this means that $C_1^{p_1}, ..., C_n^{p_n} \models A$.

Per 2.1.6 we find either

$$C_1^{p_1}, ..., C_n^{p_n} \vdash A \text{ or } C_1^{p_1}, ..., C_n^{p_n} \vdash \neg A$$

If $C_1^{p_1}, ..., C_n^{p_n} \vdash \neg A$ then per soundness $C_1^{p_1}, ..., C_n^{p_n} \models \neg A$ which is in contradiction with our model $\mathcal{M}$ therefore we can conclude the following:

$$C_1^{p_1}, ..., C_n^{p_n} \vdash A$$

for all $(p_1, ..., p_n) \in \{0,1\}^n$.

We will prove that $\vdash A$ by induction on n.
The base case $n = 0$ is trivial. Assume $n = k + 1$. For all $(p_1, ..., p_k) \in \{0,1\}^k$ we have

$$C_1^{p_1}, ..., C_k^{p_k}, B_{k+1} \vdash A \text{ and } C_1^{p_1}, ..., C_k^{p_k}, \neg B_{k+1} \vdash A.$$

Per 2.1.5 we can conclude

$$C_1^{p_1}, ..., C_k^{p_k} \vdash A$$

for all $(p_1, ..., p_k) \in \{0,1\}^k$ and per induction assumption we get $\vdash A$ as desired. □ □

Remark 2.1.8. Until now we have been working in a propositional proof systems, ignoring the variables in our language. We have done this as certain proofs -namely completeness- are a lot easier in propositional proof systems. Statements like $\exists x A$ are in general not decidable. In other words there can exist no algorithm which decides the truth of any statement $\exists x\, A$. Proofs similar to the one given above are therefore impossible for first-order logic.

In further chapters we will prove completeness of different first-order proof systems, these proofs have some clever tricks for getting around the undecidability issue.

Definition 2.1.9 (First-order $\mathcal{H}$)**.** To extend our propositional proof system $\mathcal{H}_p$ to a first-order proof system $\mathcal{H}_{\mathcal{FO}}$ we will add the following axioms:

$$\forall x\, Fx \supset Ft, \tag{$\forall \to$}$$
$$Ft \supset \exists x\, Fx, \tag{$\to \exists$}$$

For any term t.
We also add two more inference rules

$$\frac{C \supset Fb}{C \supset \forall x\, Fx} \quad (\to \forall) \qquad\qquad \frac{Fb \supset C}{\exists x\, Fx \supset C} \quad (\exists \to)$$

Where b may be any variable, with the restriction that for each of these inference rules, the variable b may not occur in the lower formula. For the remainder of this thesis we will only consider this first-order Hilbert-style system $\mathcal{H}$.

Remark 2.1.10. We can interpret these inference rules as follows: in the above proof tree we could have replaced b for any variable and it would still be a valid proof tree. Therefore we can say it is true for all/any variable x.
It is unfortunate that we have to add new inference rules. However, if we replaced these inference rules with axioms we could not enforce that b may not occur in the lower formula. As an example we could construct the following proof tree

$$\dfrac{(C \supset Fb) \supset (C \supset \forall x\, Fx) \qquad \dfrac{Fa \supset C \supset Fb \qquad (X \supset Y) \supset (Y \supset Z) \supset (X \supset Z)}{((C \supset Fb) \supset (C \supset \forall x\, Fx)) \supset (Fb \supset (C \supset \forall x\, Fx)}}{Fb \supset C \supset \forall x\, Fx}$$

Using a well-chosen C we can now prove $Fb \supset \forall x\, Fx$, which is obviously not always true.

There are different methods to solve this problem, but the rules specified above are the simplest.

2.2 Natural Deduction

In this section we will be introducing the system of natural deduction ND. This system, together with sequent calculus will be the main focus of this thesis.
Natural deduction was first introduced by Gentzen [5] and created to solve a problem Gentzen saw in existing proof-systems.

> "[..] The formalization of logical deduction, especially as it has been developed by Frege, Russell, and Hilbert, is rather far removed from the forms of deduction used in practice in mathematical proofs."
> ([5], Gentzen, Investigations into logical deduction, p288)

His stated goal was to create a system as close to actual reasoning as possible. Contrary to other systems, each formula in an ND proof is a conditional tautology, rather than an unconditional tautology as seen previously for Hilbert-style systems.
Instead of starting with axioms, we now start with assumptions, and through this use of assumptions we can replicate reasoning steps from mathematical proofs. For example we can prove by contradiction: We can assume a statement A and show it leads to a contradiction, from this we can then conclude our assumption was false.

We will be working with a modified version of natural deduction as formulated by Prawitz [3].
Prawitz made small but significant changes to the system as they allow us to introduce a normal form of proofs. Using this normal form we can infer many more properties of the natural deduction system.

2.2.1 The system of Natural Deduction

On the surface we can describe natural deduction as a proof system with no axioms and instead a rather large set of inference rules. This lack of inherent axioms will allow us to work with *assumptions*. Assumptions are used as axioms, except they can be *discharged* i.e. removed. We will call assumptions *active* if they have not been discharged.

The construction of an ND proof tree will be done as follows: We start with an assumption A. This gives us an assumption tree, with A as both its only leaf and the end-formula. Since A has not been discharged, the meaning of this tree is $A \vdash A$.
After this we apply inference rules, combining proof trees until we reach our desired result.
Several inference rules allow us to discharge assumptions of a specific shape. After an assumption is discharged, the end-formula of the proof tree is no longer dependent on it. As an example: say we have a proof tree Π with end-formula B and assumption A. If we choose to infer $A \supset B$, we can then discharge the assumption A. As this assumption is clearly no longer needed for our conclusion.

Definition 2.2.1 (Inference rules)**.** We will now formulate the inference rules for natural deduction. We write $[A]$ as discharging all formulas of shape A.

$$\frac{A \qquad B}{A \wedge B}\ (\wedge I) \qquad \frac{A \wedge B}{A}\ (\wedge E)$$

$$\frac{A}{A \vee B}\ (\vee I) \qquad \frac{A \vee B \qquad \begin{matrix}[A]\\ C\end{matrix} \qquad \begin{matrix}[B]\\ C\end{matrix}}{C}\ (\vee E)$$

$$\frac{\begin{matrix}[A]\\ B\end{matrix}}{A \supset B}\ (\supset I) \qquad \frac{A \qquad A \supset B}{B}\ (\supset E)$$

$$\frac{Fa}{\exists x\, Fx}\ (\exists I) \qquad \frac{\exists x\, Fx \qquad \begin{matrix}[Fa]\\ C\end{matrix}}{C}\ (\exists E)$$

$$\frac{Fa}{\forall x\, Fx}\ (\forall I) \qquad \frac{\forall x\, Fx}{Fa}\ (\forall E)$$

$$\frac{\begin{matrix}[\neg A]\\ \bot\end{matrix}}{A}\ (\bot_c)$$

For $\forall I$ and $\exists E$ we name a the eigenvariable of this inference.

Returning to our graph representation of proof trees in (1.3), each of these inference rule connects the end-formulae of up to three proof trees to a new node.

Definition 2.2.2 (Deduction rules)**.** For most of these inference rules, the resulting proof tree will be dependent on the assumptions of its direct sub-trees. For these inference rules our graphical notation is sufficient. We identify $\vee E$, $\supset I$, $\forall I$, $\exists E$, $\bot_c$ as *improper inference rules* as these either discharge assumptions or have constraints.
We will use a notation from Prawitz [3], this notation will accompany the inference rules specified above, explicitly stating the corresponding transformations to our assumptions.
We can explicitly notate the active assumptions and the end-formula of a proof tree Π with $\langle \Gamma, A\rangle$. In this, Γ is the set of active assumptions and A the end-formula. As an example: all assumption trees can be written as $\langle A, A\rangle$.

A deduction rule has the shape $\langle \Sigma_1, \Sigma_2, ..., \Sigma_k, \Pi\rangle$ signifying that a proof tree Π can be inferred from proof trees Σ_i. We can now formulate our deduction rules

$\vee E$ $\langle\langle \Gamma_1, A \vee B\rangle, \langle \Gamma_2, C\rangle, \langle \Gamma_3, C\rangle, \langle \Delta, C\rangle\rangle$ where $\Delta = \Gamma_1 \cup (\Gamma_2 \setminus \{A\}) \cup (\Gamma_3 \setminus \{B\})$.

$\supset I$ $\langle\langle \Gamma, B\rangle, \langle \Delta, A \supset B\rangle\rangle$ where $\Delta = \Gamma \setminus \{A\}$.

$\forall I$ $\langle\langle \Gamma, A\rangle, \langle \Gamma, \forall x\, A[x/a]\rangle\rangle$ where a does not occur in any formula in Γ.

$\exists E$ $\langle\langle \Gamma_1, \exists x A\rangle, \langle \Gamma_2, C\rangle, \langle \Delta, C\rangle\rangle$ where $\Delta = \Gamma_1 \cup (\Gamma_2 \setminus \{A[a/x]\})$ where a does not occur in $\exists x A$ or any formula in $\Gamma_2 \setminus \{A[a/x]\}$.

$\bot_c$ $\langle\langle \Gamma, \bot\rangle, \langle \Delta, A\rangle\rangle$ where $\Delta = \Gamma \setminus \{\neg A\}$ and A is not of the form $\neg B$ or $\bot$.[2]

2.2.2 Properties

Remark 2.2.3. The inference rules of ND are quite similar to the axioms in $\mathcal{H}$. All proper inference rules have equivalent axioms in $\mathcal{H}$, except for the $(\supset E)$ rule which is exactly the modus ponens rule.

[2]This restriction is not strictly necessary as our proof trees will still be sound without it. However, the restriction on $\bot_c$ is used to simplify later proofs, in which we will seperate by case on the shape of A. Furthermore this will disallow $\frac{\bot}{\bot}$. In other words we can no longer perform a useless inference of $\bot$ from $\bot$.

In practice however, these systems are very different. Working with $\mathcal{H}$ feels almost like solving a puzzle, having to formulate each step as a tautology and finding axioms to get to your desired result.
While in natural deduction, because we are able to replicate steps from mathematical proofs, it indeed feels quite natural. We can easily transfer a lot of mathematical proofs to the natural deduction system without changing much of the structure.

Theorem 2.2.4 (Soundness)**.** *The system of natural deduction ND is implicationally sound. In other words for any set of formulas Γ and any formula A we have that $\Gamma \vdash A$ implies $\Gamma \models A$.*

Proof. We will say a proof tree Σ with active assumptions in Γ and end-formula B is sound if and only if $\Gamma \models B$ We prove our theorem using induction: Clearly assumption trees are sound. For any A we have $A \models A$.

All that is left is to convince ourselves that each of the inference rules preserves truth i.e. if the sub-trees are sound then our inferred proof tree is sound as well. From this our theorem follows per induction. □

2.3 Equivalence

In this section we will prove we can *model* $\mathcal{H}$ in ND, see (1.3) for the definition.

This proof first appeared in Gentzen's ("Investigations into logical deduction [5]"). Gentzen proved the equivalence between three proof-systems: NDF which is a Hilbert-style system with different axioms from $\mathcal{H}$, ND and sequent calculus, which will be introduced later in this thesis. He did this by showing that ND models NDF, sequent calculus models ND and finally NDF models sequent calculus. We will instead directly prove the equivalence between $\mathcal{H}$ and ND. This first proof is taken from Gentzen, except as translated to our axiom schema.

Theorem 2.3.1 ($\mathcal{H}$ to ND)**.** *The proof system ND models the proof system $\mathcal{H}$. Any proof tree in $\mathcal{H}$ can be rewritten as a proof tree in ND with no active assumptions.*

Proof. Let Π a proof tree in $\mathcal{H}$, we will prove we can rewrite this to a proof tree in ND. This will be done using induction on the number of inference steps in our proof tree.

For our base case Π contains only an axiom. For each axiom we can construct a proof tree in ND with no active assumptions.

$$\dfrac{\dfrac{A^1}{B \supset A}\supset I}{A \supset B \supset A}\supset I,1 \qquad \dfrac{\dfrac{A \supset A \supset B^1 \quad A^2}{A \supset B} \quad A^3}{\dfrac{\dfrac{B}{A \supset B}\supset I,2,3}{(A \supset A \supset B) \supset (A \supset B)}\supset I,1}$$

$$\dfrac{\dfrac{A^1 \quad A \supset B^2}{B} \quad B \supset C^3}{\dfrac{\dfrac{\dfrac{C}{A \supset C}\supset I,1}{(B \supset C) \supset (A \supset C)}\supset I,3}{(A \supset B) \supset (B \supset C) \supset (A \supset C)}\supset I,2} \qquad \dfrac{\dfrac{A \supset B \supset C^1 \quad A^2}{B \supset C} \quad B^3}{\dfrac{\dfrac{\dfrac{C}{A \supset C}\supset I,2}{B \supset A \supset C}\supset I,3}{(A \supset B \supset C) \supset (B \supset A \supset C)}\supset I,1}$$

$$\dfrac{\dfrac{A \wedge B}{A}}{A \wedge B \supset A} \qquad \dfrac{\dfrac{A}{A \vee B}}{A \supset A \vee B}$$

$$\dfrac{\dfrac{\dfrac{A^1 \quad B^2}{A \wedge B}\supset I,2}{B \supset A \wedge B}\supset I,1}{A \supset B \supset A \wedge B} \qquad \dfrac{\dfrac{\dfrac{\dfrac{A \vee B^1 \quad \dfrac{A^2 \quad A \supset C^4}{C} \quad \dfrac{B^4 \quad B \supset C^5}{C}}{C}\vee E,2,4}{A \vee B \supset C}\supset I,1}{(B \supset C) \supset (A \vee B \supset C)}\supset I,5}{(A \supset C) \supset (B \supset C) \supset (A \vee B \supset C)}\supset I,3$$

$$\dfrac{\dfrac{Fa}{\exists x\, Fx}}{Fa \supset \exists x\, Fx} \qquad \dfrac{\dfrac{\forall x\, Fx}{Fa}}{\forall x\, Fx \supset Fa}$$

$$\dfrac{\dfrac{\dfrac{\neg\neg A^1 \quad \neg A^2}{\bot}\supset E}{A}\bot_c,2}{\neg\neg A \supset A}\supset I,1$$

As for our induction step, we assume the theorem is true for all sub-trees of Π. We separate by case on the final inference rule application a in Π.

Case 1) a is modus ponens.
Because modus ponens is an inference rule in ND this case is trivial.

Case 2) a is $(\exists \rightarrow)$. This means the end-formula has the shape $\exists x\, Fx \supset C$ and per induction assumption there exists a proof tree Σ of $Fa \supset C$. We can now construct the following proof tree in ND

$$\dfrac{\exists x Fx^{1} \qquad \dfrac{\overset{\Sigma}{\overset{\vdots}{Fa^{2}}} \qquad \overset{\Sigma}{\overset{\vdots}{Fa \supset C}}}{C}\,\exists E, 2}{\dfrac{C}{\exists x\, Fx \supset C}\supset I, 1}$$

Because Σ contains no active assumptions and C does not contain a this proof tree is valid.

Case 3) a is $(\rightarrow \forall)$. This means the end-formula has the shape $C \supset \forall x\, Fx$ and per induction assumption there exists a proof tree Σ of $C \supset Fa$. We can now construct the following proof tree in ND

$$\dfrac{\dfrac{C \qquad \overset{\Sigma}{\overset{\vdots}{C \supset Fa}}}{Fa}}{\dfrac{\forall x\, Fx}{C \supset \forall x\, Fx}}$$

Because C does not contain a and Σ does not have any active assumptions, this proof tree is valid.

We can now conclude that if all sub-trees of Π can be rewritten as proof trees in ND then Π can be rewritten as well.
Per induction we now find that any proof tree in $\mathcal{H}$ can be rewritten to a proof tree in ND. $\square$

2.4 ND to H

In this section we will perform the first proof in this thesis I have constructed myself. This proof was inspired by Gentzen's proof that NDF models sequent calculus. His proof was structured as a series of transformations to the entire proof tree. In this he introduced 9 *new basic sequents* e.g. axioms, and 20 new inference figures. For these reasons, I found the proof difficult to follow, and most of all it was difficult to visualize what the proof tree looked like between each of the steps. In my proof I will instead introduce a small set of lemmas in the form of inference steps we can perform. Afterwards I directly prove our theorem using induction and those lemmas.

After constructing my proof I discovered that a similar proof already exists in 'Basic Proof Theory' [6]. These proofs are similar in that we both reduce most inference rules to $\supset I$ and $\supset E$. The key difference between these proofs is that the proof in [6] uses formulas in Γ as axioms and later changes the proof-tree to remove these axioms.
My proof serves as an alternative proof, where we show more directly how to perform the same inference steps without changing previous parts of our proof-tree. Because of this change we will need additional lemmas which I have formulated myself.

Definition 2.4.1. In an ND tree, for every formula occurence A in the tree and every list of active assumptions $D_k, .., D_1$ above A, the formula $D_k \supset ... \supset D_1 \supset A$ is a tautology. We write $\{A\}$ to denote this formula, note that there are multiple possible configurations for $\{A\}$

In this section we will prove the following theorem.

Theorem 2.4.2. *The proof system $\mathcal{H}$ models ND. In particular for every proof tree Π in ND with end-formula A, there exists a proof tree Π' in $\mathcal{H}$ with end-formula $\{A\}$.*

Lemma 2.4.3 (Stronger Transitivity)**.** *If there exist proof trees Σ_1 and Σ_2 for $D_k \supset ... \supset D_1 \supset A$ and $A \supset B$ respectively, then through a series of inference steps, we can construct a proof tree Π with end-formula $D_n \supset ... \supset D_1 \supset B$. We will write this transformation as follows*

$$\dfrac{\begin{array}{cc}\Sigma_1 & \Sigma_2 \\ \{A\} & A \supset B\end{array}}{\{B\}}$$

Proof. We will prove this by constructing a hilbert-style proof tree of

$$(D_k \supset ... \supset D_1 \supset A) \supset (D_k \supset ... \supset D_1 \supset B), \tag{*k}$$

from a proof tree of $A \supset B$, for any $k \geq 0$.
We will do this using induction.
For our base case we have $k = 0$ and the proof tree Σ_2 of $A \supset B$ is sufficient.

For our induction case we will assume our claim is true for a specific $k \in \mathbb{N}$, e.g. there exists a proof tree of Π_k of (*k). We will now construct a proof tree for the case $k + 1$.
If we substitute $X = D_{k+1}, Y = (D_k \supset ... \supset A)$ and $Z = (D_k \supset ... \supset B)$ into the transitivity axiom we construct the following formula:

$$(D_{k+1} \supset (D_k \supset ... \supset A)) \supset ((D_k \supset ... \supset A) \supset (D_k \supset ... \supset B)) \supset (D_{k+1} \supset (D_k \supset ... \supset B)).$$

As the middle formula $Y \supset Z$ is exactly the end-formula of Π_k we can apply an exchange axiom and two modus ponens applications to find

$$(D_{k+1} \supset (D_k \supset ... \supset D_1 \supset A)) \supset (D_{k+1} \supset (D_k \supset ... \supset D_1 \supset B))$$

Per induction we may now conclude (*k) is true for all $k \in \mathbb{N}$.
We can now construct the proof tree Π_n, if we apply modus ponens between this and Σ_1 we find our desired result. □

Lemma 2.4.4 (Stronger Exchange)**.**
Let k and l be integers such that $k \geq 0$ and $m \geq 1$. If there exists a proof tree Σ of $D_k \supset ... \supset D_1 \supset A \supset B \supset C_m \supset ... \supset C_1$, then we can apply several inference rules to construct a proof tree Π with end-formula $D_k \supset ... \supset D_1 \supset B \supset A \supset C_m \supset ... \supset C_1$. In other words, we can exchange the placement of two adjacent assumptions, anywhere in the formula. We will notate one or more of these transformations with a double line, similar to (2.4.3)*:*

$$\cdot\ \frac{\begin{matrix}\Sigma\\ D_k \supset ... \supset D_1 \supset A \supset B \supset C_m \supset ... \supset C_1\end{matrix}}{\begin{matrix}\\ D_k \supset ... \supset D_1 \supset B \supset A \supset C_m \supset ... \supset C_1\end{matrix}}$$

Proof. We will write $C = C_m \supset ... \supset C_1$, from the exchange axiom we now find $(A \supset B \supset C) \supset (B \supset A \supset C)$. Applying (2.4.3) we get the following proof tree

$$\frac{\dfrac{\begin{matrix}\Sigma\\ \{A \supset B \supset C\}\end{matrix} \qquad (A \supset B \supset C) \supset (B \supset A \supset C)}{\{B \supset A \supset C\}}}{D_k \supset ... \supset D_1 \supset B \supset A \supset C_m \supset ... \supset C_1}$$

as desired. □

Lemma 2.4.5 ($\supset E$)**.** *If there exists a $\mathcal{H}$-proof tree Σ_1 of $D_n \supset ... \supset D_1 \supset A$ and a proof tree Σ_2 of $C_k \supset ... \supset C_1 \supset A \supset B$ then from Σ_1 and Σ_2 we can infer a proof tree Π of $D_n \supset ... \supset D_1 \supset C_k \supset ... \supset C_1 \supset B$.*

We will notate this by writing

$$\frac{\begin{matrix}\Sigma_1\\ \{A\}\end{matrix} \qquad \begin{matrix}\Sigma_2\\ \{A \supset B\}\end{matrix}}{\{B\}}$$

Proof. We can easily show this by applying k exchanges to Σ_2 and afterwards using (2.4.3).

$$\frac{\dfrac{\begin{matrix}\Sigma_1\\ \{A\}\end{matrix} \qquad \dfrac{\begin{matrix}\Sigma_2\\ C_k \supset ... \supset A \supset B\end{matrix}}{A \supset (C_k \supset ... \supset B)}}{\{C_k \supset ... \supset B\}}}{D_n \supset ... \supset D_1 \supset C_k \supset ... \supset C_1 \supset B}$$

as desired. □

Lemma 2.4.6 ($\supset I$)**.** *If there exists a $\mathcal{H}$-proof tree Σ of $\{B\}$, then from Σ we can construct a proof tree Π of $\{A \supset B\}$ where A only occurs in $A \supset B$ and not as an assumption in $\{A \supset B\}$.*

We will notate this as follows

$$\frac{\begin{matrix}\Sigma\\ \{B\}\end{matrix}}{\{A \supset B\}}$$

Proof. Using several exchanges we can transform the conclusion of Σ to $A \supset ... \supset A \supset D_k \supset ... \supset B$. In this we have separated all assumptions of the shape $[A]$ from the rest. Say we have n assumptions of this shape, if $n \neq 0$ we apply contraction $n - 1$ times and reduce this amount to either 1.
In this case we can apply k exchanges to get our desired result. If on the other hand $n = 0$, then there were no assumptions of the shape A and we can instead apply the constant axiom $B \supset A \supset B$ with (2.4.3). From this we get our desired result. □

Lemma 2.4.7 (Induction Step)**.** *Let Π a natural deduction proof tree with conclusion Y. Let α the final application of a deduction rule in Π, say $\alpha = \langle \Sigma_1, ..., \Sigma_k, \Pi \rangle$ for a $k \in \{1, 2, 3\}$. Let Σ_i have conclusion X_i. If for every $i \leq k$ there exists an $\mathcal{H}$ proof tree Σ_i' with conclusion $\{X_i\}$. Then from these proof trees Σ_i' we can now infer Π' with conclusion $\{C\}$.*

Proof. We separate by case on all inference rules. We will rewrite each of these as applications of the previous lemmas and therefore inference rules in $\mathcal{H}$.

$$\dfrac{\Sigma_1 \quad \{B\} \qquad \dfrac{\Sigma_2 \quad \{A\} \qquad A \supset B \supset A \wedge B}{\{B \supset A \wedge B\}}}{\{A \wedge B\}} \tag{$\wedge I$}$$

$$\dfrac{\Sigma \quad \{A \wedge B\} \qquad A \wedge B \supset A}{\{A\}} \tag{$\wedge E$}$$

$$\dfrac{\Sigma \quad \{A\} \qquad A \supset A \vee B}{\{A \vee B\}} \tag{$\vee I$}$$

$$\dfrac{\Sigma_1 \quad \{A \vee B\} \qquad \dfrac{\dfrac{\Sigma_2 \quad \{C\}}{\{B \supset C\}} \qquad \dfrac{\dfrac{\Sigma_3 \quad \{C\}}{\{A \supset C\}} \qquad (A \supset C) \supset (B \supset C) \supset (A \vee B \supset C)}{\{(B \supset C) \supset (A \vee B \supset C)\}}}{\{A \vee B \supset C\}}}{\{C\}} \tag{$\vee E$}$$

$$\dfrac{\Sigma \quad \{Fa\} \qquad Fa \supset \exists x\, Fx}{\{\exists x\, Fx\}} \tag{$\exists I$}$$

$$\dfrac{\Sigma_1 \quad \{\exists x\, Fx\} \qquad \dfrac{\dfrac{\dfrac{\Sigma_2 \quad \{C\}}{Fa \supset D_k \supset ... \supset D_1 \supset C}}{\exists x\, Fx \supset D_k \supset ... \supset D_1 \supset C}}{\{\exists x\, Fx \supset C\}}}{\{C\}} \tag{$\exists E$}$$

$$\dfrac{\Sigma \quad \{\forall x\, Fx\} \qquad \forall x\, Fx \supset Fa}{\{Fa\}} \tag{$\forall E$}$$

$$\dfrac{\dfrac{\Sigma \quad \{\bot\}}{\{\neg A \supset \bot\}} \qquad \neg\neg A \supset A}{\{A\}} \tag{$\bot_c$}$$

We note $\exists E$ as a rather difficult case, as we must make all dependencies explicit in order to perform $(\exists \rightarrow)$ inference rule in the $\mathcal{H}$ system. As a keen eyed reader might have noticed: We omitted $\forall I$ as this again requires us to use not just modus ponens but also the $\rightarrow \forall$ inference rule. We have $D_k \supset .... \supset D_1 \supset Fa$ where D_i contains no a. We want to transform this to $C \supset Fa$, however we cannot mirror $\exists E$ and write

$(D_k \supset ... \supset D_1) \supset Fa$ as this has an entirely different meaning. Rather than "If D_k to D_1 are true, we have Fa" we now have "If D_k to D_2 imply D_1 then Fa". Instead we must transform our statement to $(...(D_{k+1} \wedge D_k) \wedge ... \wedge D_1) \supset Fa$. As this transformation is similar to the previous induction proofs, we will leave it as a lemma (2.4.8) at the end of the section. We will now complete our proof by cases with the case $(\forall I)$

$$\dfrac{\dfrac{\dfrac{\dfrac{\Sigma_1'}{D_k \supset ... \supset D_1 \supset Fa}}{(...(D_{k+1} \wedge D_k) \wedge ... \wedge D_1) \supset Fa}}{(...(D_{k+1} \wedge D_k) \wedge ... \wedge D_1) \supset \forall x\, Fx}}{D_k \supset ... \supset D_1 \supset \forall x\, Fx}$$

as desired. □

proof of theorem(2.4.2). We prove the theorem by induction on the length of Π. Every assumption tree of A has $\{A\} = A \supset A$ and therefore we can construct the equivalent $\mathcal{H}$ proof tree:

$$\dfrac{A \supset A \supset A \qquad (A \supset A \supset A) \supset (A \supset A)}{A \supset A}$$

□

Our induction case is proven by (2.4.7). From this we can conclude our theorem.

Lemma 2.4.8 $((\forall I))$**.** *If there exists an $\mathcal{H}$ proof tree Σ of $D_k \supset ... \supset D_1 \supset X$ then from this we can infer a proof tree of $(...(D_{k+1} \wedge D_k) \wedge ... \wedge D_1) \supset X$ and vice versa. We will notate this transformation with a double line. We will prove this by induction. The base case $k = 0$ and also $k = 1$ are trivial.*
As for our induction case: We will use shorthand to improve readability of the proof tree: let $A = D_{k+1}, B = D_k$ and $C = D_{k-1} \supset ... \supset X$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\Sigma}{\dfrac{A \wedge B \supset A \qquad A \supset B \supset C}{A \wedge B \supset B \supset C}} \qquad (A \wedge B \supset B \supset C) \supset (B \supset A \wedge B \supset C)}{A \wedge B \supset B \qquad\qquad B \supset A \wedge B \supset C}}{A \wedge B \supset C}}{(...(D_{k+1} \wedge D_k) \wedge ... \wedge D_1) \supset X}$$

and the reverse:

$$\dfrac{\dfrac{\dfrac{\Pi}{(...(D_{k+1} \wedge D_k) \wedge ... \wedge D_1) \supset X}}{A \supset B \supset A \wedge B \qquad\qquad (A \wedge B) \supset C}}{A \supset B \supset C}$$

We have demonstrated we can transform any inference rule in ND to an equivalent form in $\mathcal{H}$, therefore per induction we find that we can model any ND proof in $\mathcal{H}$.

3 Sequent Calculus

In this chapter we will introduce the sequent-calculus proof system. Sequent calculus was introduced together with natural deduction by Gentzen in *Investigations into logical deduction* [5].
Sequent calculus can be viewed as a step between Hilbert-style systems and natural deduction. In proving that Hilbert style systems model natural deduction, we first want to transform each formula to an unconditional tautology. Our method for this was recursively writing $A \supset ...$ for every assumption A. Sequent calculus instead writes each of these statements as a *sequent* 3.1.1. These sequents split the unconditional tautology and the assumptions needed, written in one statement.
We will use a modified version of sequent calculus which is very similar to the definitions given in [Buss, ch 1.2, [1]]. Most of this chapter has been taken from or inspired by Buss.

3.1 Definition

Definition 3.1.1 (Sequents)**.**
Sequent calculus considers *sequents* instead of formulas as the nodes of its proof tree. Instead of a single formula, sequents contain a left- and a right sequence of formulas e.g. $A_1, ..., A_k$, we call such a sequence a *cedent*. Specifically a sequent has the form

$$A_1, ..., A_k \to B_1, ..., B_l.$$

In this sequent the right cedent is named the *succedent* of our sequent and contains our conclusion formulas. The left cedent, named the *antecedent* contains all of our assumptions. This is in contrast to natural deduction where all assumptions are contained in the leaves of a proof tree.

Remark 3.1.2. The meaning of a sequent is that the conjunction of the antecendent implies the disjunction of the succedent. In other words, our sequent is equivalent to

$$\bigwedge_{i=0}^{k} A_i \supset \bigvee_{i=0}^{l} B_i.$$

Sequents are an in-between step between conditional and unconditional tautologies. This combines advantages of both. Conditional tautologies allow a lot more freedom in reasoning. But because we make all dependencies explicit we can consider each sequent on its own: Instead of taking into account the entirety of the proof tree above it.

From our definitions, we find that smaller cedents give us 'stronger' statements. A statement clearly is stronger if you need less assumptions, or if your conclusion is more precise.

For our interpretation we have to consider a few edge cases however. We define the empty conjunction to be true. This means that every formula A can be rewritten as a sequent $\to A$, which is equivalent to $\top \supset A$. We define the empty disjunction to be false, as such the empty sequent $\to$ is equivalent to $\top \supset \bot$ which is false.

Definition 3.1.3 (Weak Structural Rules)**.** We have split our propositional rules into *weak-* and *strong-inference rules*. For each, we use Γ, Π, Δ and Λ for arbitrary cedents. We have the following weak structural rules:

$$\frac{\Gamma, A, B, \Pi \to \Delta}{\Gamma, B, A, \Pi \to \Delta} \quad \textit{Exchange:left} \qquad \frac{\Gamma \to \Delta, A, B, \Lambda}{\Gamma \to \Delta, B, A, \Lambda} \quad \textit{Exchange:right}$$

$$\frac{A, A, \Gamma \to \Delta}{A, \Gamma \to \Delta} \quad \textit{Contraction:left} \qquad \frac{\Gamma \to \Delta, A, A}{\Gamma \to \Delta, A} \quad \textit{Contraction:right}$$

$$\frac{\Gamma \to \Delta}{A, \Gamma \to \Delta} \quad \textit{Weakening:left} \qquad \frac{\Gamma \to \Delta}{\Gamma \to \Delta, A} \quad \textit{Weakening:right}$$

Remark 3.1.4. These weak inference rules do not infer something new, but instead involve positioning of formulas. Because these rules are less significant, we will often write a double line as shorthand for a series of weak structural rules. For example we might write

$$\frac{A \to A}{A, \Gamma \to \Delta, A}$$

for inserting another sequent into our axiom. In reality this would involve a long series of weakenings and exchanges. But because it is intuitively quite a simple transformation we will write it as such.

Definition 3.1.5 (Propositional Rules)**.** For our strong inference rules, we will first discuss the propositional inference rules. Shortly after this definition, we will also introduce our first-order inference rules.

$$\frac{A, B, \Gamma \to \Delta}{A \wedge B, \Gamma \to \Delta} \wedge\textit{:left} \qquad \frac{\Gamma \to \Delta, A \qquad \Gamma \to \Delta, B}{\Gamma \to \Delta, A \wedge B} \wedge\textit{:right}$$

$$\frac{A, \Gamma \to \Delta \qquad B, \Gamma \to \Delta}{A \vee B, \Gamma \to \Delta} \vee\textit{:left} \qquad \frac{\Gamma \to \Delta, A, B}{\Gamma \to \Delta, A \vee B} \vee\textit{:right}$$

$$\frac{\Gamma \to \Delta, A \qquad B, \Gamma \to \Delta}{A \supset B, \Gamma \to \Delta} \supset\textit{:left} \qquad \frac{A, \Gamma \to \Delta, B}{\Gamma \to \Delta, A \supset B} \supset\textit{:right}$$

Remark 3.1.6. We have made a specific change to the sequent calculus system introduced by Buss. Instead of considering the symbol $\neg$ a connective in our language we instead see it as a shorthand. Namely, $\neg A$ is short for $A \supset \bot$. Because of this, our system does not contain $\neg : left$ and $\neg : right$ rules. But as we still want to perform the same operations, we instead allow for *initial sequents* (see (3.1.11) for the definition) of the shape $\bot \to$. We will demonstrate how to perform the $\neg$-rules in our version of sequent calculus.

$$\frac{\Gamma \to \Delta, A}{\neg A, \Gamma \to \Delta} \neg\textit{:left} \qquad \frac{\Gamma \to \Delta, A \qquad \dfrac{\bot \to}{\bot, \Gamma \to \Delta}}{\neg A, \Gamma \to \Delta} \supset\text{: } left$$

$$\frac{A, \Gamma \to \Delta}{\Gamma \to \Delta, \neg A} \neg\textit{:right} \qquad \frac{\dfrac{A, \Gamma \to \Delta}{A, \Gamma \to \Delta, \bot}}{\Gamma \to \Delta, \neg A} \supset\text{: } right$$

Note that that transformations still each involve just one strong inference. In [1] the amount of strong inferences in a proof tree is used as its length, this change will therefore not invalidate any theorems on upper bounds of proof tree size.

Definition 3.1.7. We have two more inference rules if we want to extend our system to first-order logic.

$$\frac{Ft, \Gamma \to \Delta}{\forall x\, Fx, \Gamma \to \Delta} \forall : left \qquad \frac{\Gamma \to \Delta, Fb}{\Gamma \to \Delta, \forall x\, Fx} \forall : right$$

$$\frac{Fb, \Gamma \to \Delta}{\exists x\, Fx, \Gamma \to \Delta} \exists : left \qquad \frac{\Gamma \to \Delta, Ft}{\Gamma \to \Delta, \exists x\, Fx} \exists : right$$

Where t may be any term and b must be a free variable which does not occur in the lower sequent. Without this restriction we could always create sequents like $\exists x\, Fx \to \forall x\, Fx$. We name b the eigenvariable of the $\forall : right$ or $\exists : left$ inference.

Definition 3.1.8 (Cut rule)**.** Our final strong inference rule is the cut rule:

$$\frac{\Gamma \to \Delta, C \qquad C, \Gamma \to \Delta}{\Gamma \to \Delta}$$

The cut-rule is sort of an outlier in our system. Allowing for cuts creates formulas without descendants, which causes problems for the sub-formula principle. However cuts can significantly decrease the length of proof trees. Furthermore, cuts are actually required to make our system implicationally complete.

In chapter (4) we will demonstrate how to perform cut-elimination on a proof tree, this will remove any unnecessary cuts and gives us a modified sub-formula principle.

Definition 3.1.9. In order to facilitate our discussion of sequent calculus we will introduce some terminology, relating to a formula's position in a proof tree. These correspond to the definitions in [[1] p.12]. Every inference rule has a *principal formula* which we define as the formula in the lower sequent that is not in Γ or Δ (or Π or Λ). Every inference rule has one or more *auxiliary formulas* which are the formulas A and B (or F), occurring in the upper sequents of the inference. The formulas which occur in the cedents Γ, Δ, Π or Λ are called *side formulas*. Auxiliary formulas are the immediate ancestors of the principal formula, while the ancestor of a side formula is the same formula at the same position in the upper sequent. Note that all inference rules do not change the position (relative to $\rightarrow$) of side formulas in the sequent. We define *ancestors* as the reflexive and transitive closure of immediate ancestors. If a formula A is the ancestor of a formula B then we define B as the *descendant* of A. We similarly define *immediate-descendants*. Finally two formulas are each other's *direct-ancestor or -descendant* if they also have the same shape, e.g side formulas are direct-descendants of their upper counterpart.

Remark 3.1.10. An interesting property of sequent calculus compared to natural deduction. Is that we only have *introduction rules* and no elimination rules. Instead of eliminating symbols from our conclusions, we introduce symbols in our assumptions, which has a similar effect.
A consequence of this is that for any formula A, all its ancestors are also sub-formulas of A. We call this the *sub-formula principle*.

Definition 3.1.11 (proofs)**.** The leaves of our proof trees are called *initial sequents*. These are our axioms, we will write $\Gamma \vdash S$ if there exists a proof tree with end-sequent S and initial sequents in Γ. We have an initial sequent which is always present. We can always use $A \rightarrow A$ as an initial sequent for any atomic formula A. We contrary to Buss [1] also freely allow for initial sequents of the shape $\bot \rightarrow$.

We will often want to talk about formulas instead of sequents however. Each formula A has an equivalent sequent $\rightarrow A$, i.e. $\top \supset A$. Using this, we may write $\Gamma \vdash A$ for $\Gamma \vdash \rightarrow A$. Similarly we may write $\Pi \vdash A$ for a set of formulas Π if there exists a proof tree of $\rightarrow A$ with initial sequents $\rightarrow C_i$ for $C_i \in \Pi$.

Theorem 3.1.12 (Soundness)**.** *The sequent calculus proof-system is implicationally sound, namely if* $\Pi \vdash A$ *then* $\Pi \vDash A$*. For any set of formulas* Π *and formula* A*.*

Proof. This easily follows from the fact that every inference rule 'preserves truth'. We will investigate a few examples, in each of these cases we assume $\bigwedge \Gamma$ is true, as the result is trivial otherwise:

- $(\wedge : right)$ We can seperate two cases, either a formula in Δ is true, or not. In the first case the lower sequent is true. In the second case, because the upper sequents are true it follows that both A and B are true, from which we may conclude the lower sequent.
- $(\supset : right)$ If A is true then the lower sequent is obviously true per upper sequent. If A is false then $A \supset B$ is true from which follows the lower sequent.
- $(\supset : left)$ From the upper left sequent we seperate two cases. Either A is false, in which case $\Gamma \rightarrow \Delta$. Or A is true, and per $A \supset B$ and $B, \Gamma \rightarrow \Delta$ we find our lower sequent is true.
- $(\bot_c)$ this is clear as $\bot$ is always false.
- The inference rules $\forall : left$ and $\exists : right$ both result in weaker versions of their upper sequent.
- For both $\forall : right$ and $\exists : left$ we can replace every occurrence of b in the proof tree above with any constant and the proof will still be valid. From which follows the lower sequent.

□

Theorem 3.1.13 (Consistency)**.** *The sequent calculus proof-system is consistent, namely we cannot construct a proof tree of* $\rightarrow \bot$*.*

Proof. Assume there existed a proof tree Π of $\rightarrow \bot$. Per sub-formula principle Π may only contain formulas of the shape $\bot$. Therefore all initial sequents either have the shape $\bot \rightarrow \bot$ or $\bot \rightarrow$.

We cannot use propositional rules in our proof tree, because these introduce connectives while $\bot$ contains no connectives. As there do not exist any weak inference rules with which we can empty the antecendent, it follows that we cannot possibly construct such a proof tree Π. □

3.2 Completeness

In this section we will show that sequent calculus with the cut-rule is implicationally complete. This proof is again from [1], I have rewritten the proof in order to both give a different perspective and make certain steps more explicit.

Theorem 3.2.1 (Completeness)**.** *The sequent calculus proof system is implicationally complete. If Γ is a set of formulas and A a formula such that $\Gamma \vDash A$. Then we have $\Gamma \vdash A$ in sequent calculus. Specifically there now exists a proof tree Σ with initial sequents $\to C_k$ for C_k in Γ. and end-sequent $\to A$. Furthermore Σ is free-cut free, i.e. Σ contains only anchored cuts and no free-cuts.*

Lemma 3.2.2. *We claim a language L is always countable, namely we can enumerate a sequence $\langle A_i, t_i\rangle$ such that for every formula B, term s and integer n there exists an integer k such in the first k elements of our sequence the element $\langle B, s\rangle$ occurs n times.*

Proof. Because every term and formula is a finite sequence of symbols and we have a countable amount of symbols, it follows we can easily enumerate each. We then create a sequence $L_i := \langle A_i, t_i\rangle$ using diagonal enumeration in which every formula- and term combination occurs once. Finally we diagonally enumerate this sequence with itself to get a sequence (L_i, L_k) and map each tuple to its first element. This will give us our desired sequence in which each L_i occurs infinitely many times. □

Lemma 3.2.3. *If $\Gamma \vDash A$ then there exists a proof tree Σ with only axioms of the shape $B \to B$ and end-formula $C_1, ..., C_k \to A$ for formulas $C_1, ..., C_k \in \Gamma$.*

Proof. We start with $\to A$ and construct our proof tree backwards. We consider a set P of active sequents. A leaf in our proof tree is considered active as long as it does not admit a formula D in its antecendent, such that either D occurs in the succedent as well or D has shape $\bot$. These sequents are not considered active as we can easily infer them. We will now specify an algorithm which will construct our desired proof tree, we will show that the algorithm will halt if there exists a proof tree of $C_1, ..., C_k \to A$.

Let $\langle A_i, t_j\rangle$ an enumeration of all formulas and terms as given in (3.2.2).

Loop: Let $\langle A_i, t_j\rangle$ be the next pair in the enumeration.

Step 1): If A_i is in our set of axioms Π then we will add A_i to every sequent in our proof tree. Namely if $\Gamma' \to \Delta'$ is a sequent in our proof tree we will replace it with $\Gamma', A_i \to \Delta'$. The placement of A_i ensures we do not influence any other inference rules.

Step 2): If A_i is atomic we do nothing this step. Otherwise we will separate by cases on the outer connective of A_i.

2a): If A_i has the shape $B \wedge C$ then every active leaf of the shape $\Gamma', B \wedge C, \Gamma'' \to \Delta$ will be replaced by the following inference

$$\dfrac{\dfrac{B, C, \Gamma', B \wedge C, \Gamma'' \to \Delta}{B \wedge C, \Gamma', B \wedge C, \Gamma'' \to \Delta}}{\Gamma', B \wedge C, \Gamma'' \to \Delta}$$

And similarly every active leaf of the shape $\Gamma' \to \Delta', B \wedge C, \Delta''$ will be replaced with

$$\dfrac{\dfrac{\Gamma' \to \Delta', B \wedge C, \Delta'', B \qquad \Gamma' \to \Delta', B \wedge C, \Delta'', C}{\Gamma' \to \Delta', B \wedge C, \Delta'', B \wedge C}}{\Gamma' \to \Delta', B \wedge C, \Delta''}$$

2b-2c): The cases where A_i has outer connective $\supset$ or $\vee$ are dual to 2b. We find the corresponding propositional inference rule and keep both the auxiliary formulas and the principal formula in the upper sequents.

2d): If A_i has the shape $(\forall x)Fx$ then we replace every active leaf of the shape $\Gamma' \to \Delta', (\forall x)Fx, \Delta''$ are replaced with

$$\dfrac{\dfrac{\Gamma' \to \Delta', (\forall x)Fx, \Delta'', Fa}{\Gamma' \to \Delta', (\forall x)Fx, \Delta'', (\forall x)Fx}}{\Gamma' \to \Delta', (\forall x)Fx, \Delta''}$$

where a is a free variable that has not occurred in the proof tree yet. Any sequents of the shape $\Gamma', (\forall x)Fx, \Gamma'' \to \Delta$ are replaced with

$$\dfrac{\dfrac{Ft_i, \Gamma', (\forall x)Fx, \Gamma'' \to \Delta}{(\forall x)Fx, \Gamma', (\forall x)Fx, \Gamma'' \to \Delta}}{\Gamma', (\forall x)Fx, \Gamma'' \to \Delta}$$

Note this is and 2e the only cases where we use t_i.

2e): The case where A_i has shape $\exists x\, Fx$ is dual to 2d.

3): If there are no remaining active sequents in P, we exit the loop.

If this algorithm halts, every leaf admits a formula D such that either

1. The formula D occurs in both its succedent and antecendent. We can easily infer this sequent from the axiom $D \to D$ using weak-inferences.
2. The formula D has shape $\bot$. We can easily infer this from the axiom $\bot \to$.

This gives us a valid proof of our desired end-sequent.

If on the other hand the algorithm never halts. Then we instead construct an infinite proof tree. Which has at least one infinite branch π. We will now construct a model $\mathcal{M}$ where every interpretation of every constant is just the string of the variable itself $c^M = c$. But we also add interpretations for free variables $a^M = a$, $\mathcal{M}$ and similarly for terms we add objects $f^M(a_1, ..., a_k) := f(a_1, ..., a_k)$. Besides this $\mathcal{M}$ contains no further objects. For every relational symbol R we define R^M as containing only the set of tuples $(a_1, ..., a_k)$, where $R(a_1, ..., a_k)$ occurs in the antecendent of a sequent in Π.

We claim that in this model every formula in the antecendent is true, while every formula in the succedent is false. We will prove this claim using induction. Let A occur in π.

Base Case: If A atomic and occurs in an antecendent in our branch then per definition A is true. Note that A cannot have shape $\bot$.
If on the other hand A occurs in the succedent, then it does not occur in the antecendent and is therefore per definition false.

Induction case: We separate on the shape of A.

a): The formula A has shape $B \wedge C$. If A occurs in the antecendent of π then as can be seen in step 2a both B and C occur in the antecendent of π, per induction both B and C are true. Therefore A is indeed true.
If A occurs in the succedent of π then either B or C will occur in the succedent as well, per induction either B or C will be false, from which we find that A is false.

b): The case where A has shape $B \vee C$ is dual to case a).

c): The formula A has shape $B \supset C$. This case is similar to a) but still interesting as ancestors occur on a different side of the sequent than their descendants. We did not write out this case for 2c) but it is easily found by applying the $\supset$:-left and -right rules to 2a.
If $B \supset C$ occurs in the succedent of π, then B occurs in the antecendent and C in the succedent as

well. Per induction it follows that B is true and C is false, therefore $B \supset C$ is false.
If $B \supset C$ occurs in the antecendent of π then either B occurs in the succedent of π or C occurs in the antecendent. In either case we can easily see that $B \supset C$ is true.

d): The formula A has shape $(\forall x)Fx$. If $(\forall x)Fx$ occurs in the antecendent of π then Ft_i occurs in the antecendent for every t_i. Therefore we indeed find that Fx is true for all x.
If on the other hand $(\forall x)Fx$ occurs in the succedent, then the pseudo-formula Fa occurs in the succedent as well. Pseudo-formulas arent considered true or false in models. But as we introduced interpretations for every free variable. For every pseudo-formula Fa in our proof tree can instead consider the truth of Fc_i where c_i is a new constant with interpretation a^M. The rest of our proof still remains valid on our definition of truth for pseudo-formulas. We can therefore perform induction and find that Fa is *false* and therefore $(\forall x)Fx$ is false.

e): The case where A has shape $(\exists x)Fx$ is dual to d).

Per induction we find that indeed in our model every formula in the antecendent of π is true, while every formula in the succedent of π is false. In our constructed model, A is false while every $C_i \in \Gamma$ is true as they occur in the succedent and antecendent of π respectively. This is in contradiction with $\Gamma \models A$. Therefore it follows that our algorithm indeed does halt. In this model, $\Gamma \to \Delta$ is obviously false while Π is true. Oh no □

Proof of Completeness: Theorem 3.2.1 now easily follows. We will show that we can transform $C_n, ..., C_1 \to A$ to $C_{n-1}, ..., C_1 \to A$ from which our theorem follows per induction. Let $n \geq 1$, we add an initial sequent $\to C_n$ as follows:

$$\dfrac{\dfrac{\to C_n}{C_{n-1}, ..., C_1 \to A, C_n} \qquad \begin{matrix}\Sigma \\ C_n, ..., C_1 \to A\end{matrix}}{C_{n-1}, ..., C_1 \to A}$$

per induction we can now transform Σ to a proof tree of $\to A$, as desired. □

4 Cut Elimination

4.1 Free-Cut elimination

A significant part of the theory of the sequent calculus centers on removing cut inferences from proofs. In this chapter we will first demonstrate free-cut elimination theorem from Buss. [1]. Then we will demonstrate a subtle error in this proof and finally we will show how we can correct this error. This error has already been identified and fixed [2]. I developed my proof independently and as such it is my own and serves as an alternative correction that keeps more of the original proof intact. As with everything in this thesis, I have restructured the proof from as it was presented in [1].

Definition 4.1.1 (Anchored and free-cuts)**.** A formula is *anchored* if and only if it has a direct-ancestor which occurs in an initial sequent. Let α an application of the cut-rule which cuts a formula C.

- If C is atomic, then we say that α is *anchored* if and only if C is anchored in both upper sequents.
- If C is not atomic, then we say that α is *anchored* if and only if C is anchored in at least one upper sequent.

A cut which is not anchored is instead a *free-cut*.

Remark 4.1.2. We will soon show how we can remove all free-cuts from a proof tree. Firstly we will demonstrate a process with which we can remove a special kind of free-cut: weakly introduced cuts. This will be necessary as removing weakly-introduced cuts increases the number of unanchored cuts.

Definition 4.1.3 (Weakly Introduced)**.** Most formula occurrences either have direct-ancestors in initial sequents, or direct ancestors which occur as principal formulas in strong inferences. If neither of these cases hold then a formula is *weakly introduced*.

In other words a formula is weakly introduced if its ancestors have all been introduced by weakenings. A cut is weakly-introduced if either of its auxiliary formulas is weakly-introduced.

Lemma 4.1.4 (Weak-cut free proof)**.** *We can transform any proof tree Σ to a proof tree Σ' with the same end-sequent and in which no weakly introduced formulas are cut.*

Proof. Say Σ contains the following cut

$$\dfrac{\begin{array}{cc} Q & R \\ \Gamma \to \Delta, C & C, \Gamma \to \Delta \end{array}}{\Gamma \to \Delta}$$

where C is weakly introduced in Q.

Our transformation is as follows: We remove all direct ancestors of C in Q. After this transformation a number of weak inference rules might instead do nothing. This is because the principal and auxiliary formulas may have been removed. The removal of these inferences is trivial. This gives us a valid proof tree Q' of $\Gamma \to \Delta$. The case where C is weakly introduced in R is dual to our transformation of Q.

As this transformation removes significant portions of our proof tree, we might cause other formulas to become unanchored or even weakly-introduced. A formula might have only one direct ancestor in an initial sequent, which is located in R, by removing this sub-tree R this formula will then become unanchored.

Though we might increase instead of decrease the amount of weakly-introduced cuts, we note that in each transformation the **total** number of cuts decreases by at least one. Performing induction on this value, we find that we can remove all weakly-introduced cuts from a proof tree. □

Definition 4.1.5 (Free-variable normal form)**.** We define a proof tree Σ to be in *free variable normal form* if and only if every free variable occurs in an $\forall$:right or $\exists$:left inference only once and appears only in sequents above this inference.

Any proof can be transformed to a proof in free variable normal form by only renaming variables.

Remark 4.1.6. We will demonstrate how we can remove all remaining free-cuts from a proof tree. The idea behind this proof is that we can decrease the complexity of each cut, instead of cutting a formula $A \wedge B$ we perform cuts on both A and B. After continuously applying this transformation we eventually remove all free-cuts.
We will prove this using induction, but we need to take special care to show that each transformation makes *progress* to our end-goal. We cannot say that that each step reduces the amount of free-cuts, or even that it reduces the size of our proof tree, as each of these will increase during our proof.

Instead we will be performing induction on the depth of our cut-formulas. And formulate our proof such that it never increases.

Definition 4.1.7 (depth)**.**
The *depth* of a formula A is defined as the height of the tree representation of the formula. We can inductively define depth as:

- $dp(A) = 0$ for A atomic.
- $dp(A \wedge B) = dp(A \vee B) = dp(A \supset B) = max\{dp(A), dp(B)\} + 1$
- $dp(\forall x\, Ax) = dp(\exists x\, Ax) = dp(A) + 1$.

Remark 4.1.8. We will soon demonstrate the error in the cut elimination proof. The lemma below is valid but does not consider weakly-introduced cuts as we have removed all weakly introduced cuts beforehand. We claim however that it is possible to create new weakly introduced cuts during the induction process.

This would cause a problem for our proof, because as buss mentioned [1]: removing weakly introduced cuts might unanchor other cuts.

This would increase our induction value and therefore not allow us to perform induction.

Lemma 4.1.9. *Let Σ a proof tree in free variable normal form. Such that Σ has end-sequent $\Gamma \to \Delta$ and is of the following shape*

$$\dfrac{\dfrac{Q}{\Gamma \to \Delta, C} \quad \dfrac{R}{C, \Gamma \to \Delta}}{\Gamma \to \Delta}$$

where C is free and not weakly-introduced. Furthermore C has depth d and all other free-cuts in Σ have a depth strictly less than d.

*We can now transform Σ to a proof tree Σ' with the same end-sequent, such that Σ' contains only free-cuts with depth less than d. Furthermore, all formulas in $\Gamma \to \Delta$ which were anchored or **not** weakly introduced, still have these properties in Σ'.*

Proof. We separate by case on the shape of C:

(*a*) Suppose C has shape $A \wedge B$. We will transform our proof to instead perform cuts on A and B. This decreases the depth of our cut.

We will transform R to R' by replacing every sequent $\Pi \to \Lambda$ with $\Pi^-, A, B \to \Lambda$, where Π^- is equal to Π except with all direct-ancestors of $A \wedge B$ removed. This will invalidate some parts of our proof tree however, specifically initial sequents $\Pi \to \Lambda$ will be transformed to $\Pi, A, B \to \Lambda$. We fix this by instead inferring this sequent from $\Pi \to \Lambda$. Note that no direct ancestors of $A \wedge B$ occur in initial sequents, as $A \wedge B$ is not anchored.

Similarly, some $\wedge : left$ inferences will be transformed to

$$\dfrac{A, B, \Pi^-, A, B, \to \Lambda}{\Pi^-, A, B \to \Lambda}$$

We can solve this by adding exchanges and contractions to correctly remove A and B.

We will similarly transform Q to Q_X by replacing every sequent $\Pi \to \Delta$ with $\Pi, \Gamma \to \Lambda, \Delta^-$ where Δ^- is equal to Δ but with all direct ancestors of $A \wedge B$ removed. Some $\wedge : right$ inferences will now fail, a valid inference

$$\frac{\Pi \to \Delta, A \qquad \Pi \to \Delta, B}{\Pi \to \Delta, A \wedge B}$$

can become

$$\frac{\Pi, \Gamma \to \Delta, \Lambda^-, A \qquad \Pi, \Gamma \to \Delta, \Lambda^-, B}{\Pi, \Gamma, \to \Delta, \Lambda^-}$$

We can resolve this by using our transformed proof tree R'. Our inference rule will be replaced by the following construction

$$\dfrac{\dfrac{\Pi, \Gamma \to \Delta, \Lambda^-, B}{\Pi, \Gamma \to \Delta, \Lambda^-} \qquad \dfrac{\dfrac{\Pi, \Gamma \to \Delta, \Lambda^-, A \qquad A, B, \Gamma \to \Delta}{B, \Pi, \Gamma \to \Delta, \Lambda^-}\, R'}{\Pi, \Gamma \to \Delta, \Lambda^-}}{\Pi, \Gamma \to \Delta, \Lambda^-}$$

this now makes Q' a valid proof tree. With end-sequent $\Gamma, \Gamma \to \Delta, \Delta$. We now only need to perform contractions in order to transform this to a proof tree of $\Gamma \to \Delta$.

Because we took care to not discard any branches of our proof tree: all formulas in $\Gamma \to \Delta$ which were anchored or not weakly introduced in Σ will still have these properties in Σ'.

$(b), (c)$ The case where C has shape $A \vee B$ or $A \supset B$ are dual to (a).

(d) Suppose C has shape $\forall x\, Fx$. Let r a term in L we will show we can transform Q to a proof tree Q_r with end-sequent $\Gamma \to \Delta, F(r)$. We do this by replacing all direct-ancestors of C with $F(r)$. This will invalidate some inference applications however. Some $\forall : right$ inferences will now be invalid however and might look as follows

$$\frac{\Pi_i \to \Lambda_i, F(a_i)}{\Pi_i \to \Lambda_i, F(r)}$$

We can easily solve this by replacing a_i every with r in the proof tree. This transformation is valid as our proof is in free-variable normal form, because of this a_i (or r) is never used in other $\forall$:right or $\exists$:left inferences.

We transform Q by replacing every sequent $\Pi \to \Lambda$, with $\Pi, \Gamma \to \Delta, \Lambda^-$. Where Λ^- is Λ but with all direct ancestors of C removed. This will invalidate some $\forall$:left inferences and give them the following shape

$$\frac{F(t_i), \Pi, \Gamma \to \Delta, \Lambda^-}{\Pi, \Gamma \to \Delta\Lambda^-}$$

We can solve this with the following construction

$$\dfrac{\dfrac{Q_{t_i}}{\Gamma \to \Delta, F(t_i)} \qquad F(t_i), \Pi, \Gamma \to \Delta, \Lambda^-}{\Pi, \Gamma \to \Delta, \Lambda}$$

This will give us R' with end-sequent $\Gamma, \Gamma \to \Delta, \Delta$. After some contractions we will have our desired result.

(e) The case where C has the shape $\exists x\, Fx$ is dual do case d).

(f) Suppose C has shape A for A atomic. We will transform Q to Q' by replacing every sequent $\Pi \to \Lambda$ with $\Pi, \Gamma \to \Delta, \Lambda^-$, where Λ^- is equal to Λ but with all direct ancestors of C removed. This transformation will invalidate a few initial sequents however. Initial sequents of the shape $B \to B$ for B not a direct ancestor of C, will now look like $B, \Gamma \to \Delta, B$. This is not a valid initial sequent but can easily be inferred from $B \to B$ using weak inference rules. Initial sequents of the shape $A \to A$ will now have the shape $A, \Gamma \to \Delta$. This is precisely the end-sequent of R, we therefore copy this entire proof tree on top of Q'. This will give us a valid proof tree of $\Gamma, \Gamma \to \Delta, \Delta$. Again with several weak transformations we get our desired result.

□

Theorem 4.1.10 (free-cut elimination)**.** *Let S a sequent and Γ a set of sequents. If Σ is a proof tree of $\Gamma \vdash S$ then there exists a proof tree Σ' of $\Gamma \vdash S$ which contains no free-cuts.*

Proof. We prove this by applying induction twice. First we remove all weakly-introduced cuts from Σ, using lemma 4.1.4. Next we put our proof in free-variable normal form.

We define Σ_n as the result of this transformation, where n is the maximum free-cut depth in the proof tree. We claim that for any $k \geq 0$ we can transform a proof tree Σ_{k+1} to a proof tree Σ_k of $\Gamma \vdash S$ which has maximum free-cut depth k.
Say Σ_{k+1} has l cuts of depth $k+1$. We can now find a sub-tree Π of Σ_{k+1} containing only one cut of depth $k+1$, which is located at its final inference. Per lemma 4.1.9, we can remove this cut without introducing any new free-cuts of depth greater than k. With induction we can easily see we can remove all l cuts. With which we get our desired proof tree Σ_k. Using induction a second time we find we can reduce our proof tree Σ_n to a proof tree Σ_0 which contains no free-cuts. □

Remark 4.1.11 (The error)**.** We will now demonstrate an example in which the error might occur: Let Σ have the following shape

$$\dfrac{\dfrac{\dfrac{Q_1}{\Gamma \to \Delta, A} \quad \dfrac{\dfrac{Q_2}{\Gamma \to \Delta}}{\Gamma \to \Delta, B}\, Weakening : right}{\Gamma \to \Delta, A \wedge B} \qquad \dfrac{R}{A \wedge B, \Gamma \to \Delta}}{\Gamma \to \Delta,}$$

we will assume that $A \wedge B$ is not weakly introduced in R and can clearly see that it is not weakly-introduced in Q. After applying the cut elimination step from (4.1.9) we will have the following transformed proof tree

$$\dfrac{\dfrac{\dfrac{Q_2}{\Gamma \to \Delta}}{\Gamma \to \Delta, B}\, Weakening : right \qquad \dfrac{\dfrac{Q_1}{\Gamma \to \Delta, A} \quad \dfrac{R'}{A, B, \Gamma \to \Delta}}{B, \Gamma \to \Delta}\, \text{Cut}}{\Gamma \to \Delta,}\, \text{Cut}$$

We can see in this example that the final cut inference is weakly-introduced. Removing this cut inference will involve removing Q^2 and R', this might unanchor some cuts which in turn might increase our induction value and therefore make this proof invalid.

Remark 4.1.12. It should be of note however that the induction step is still valid. If we change our induction value we could still prove the cut elimination theorem. This would make our proof a bit more difficult and increase the upper bounds on the size of our proof tree by a lot potentially.

4.2 Formula reduction

In this section we demonstrate my first attempt at solving the error mentioned in the previous section. I have constructed a method to remove 'weakly connected' cuts before we perform the proof of (4.1.9), this prevents the problem mentioned in the previous section. However when writing the final part of my proof demonstrating how we would no longer introduce new weak cuts I realised that this was not the case as there were more edge cases I had not considered.

I have chosen to leave this proof in as it is an interesting construction which could in the future be used for different proofs.

Definition 4.2.1 (Formula-node)**.** Let C a formula, we can visualise all sub-formulas of C as a tree, as an example: If C is the formula $(\forall x\,(A \wedge B)) \vee (A \wedge B)$ we can write out it's formula-tree as

$$\dfrac{\dfrac{\dfrac{{}^1A \qquad B}{A \wedge B}}{\forall x\,(A \wedge B)} \qquad \dfrac{{}^2A \qquad B}{A \wedge B}}{(\forall x\,(A \wedge B)) \vee (A \wedge B)}$$

We define a *formula-node* of C as a node in this tree. The difference between a formula-node and a sub-formula is that a formula-node differentiates in the specific placement of the sub-formula within C. In other words the left formula-node ${}^1A$ and the right formula-node ${}^2A$ are not considered to be equal.

Each non-atomic formula D has one or two immediate sub-formulas. In the obvious way we define one as the left-sub-formula and the other as the right-sub-formula. If D only has one immediate sub-formula we will define it as the left-sub-formula.

We can now uniquely find each formula-node by a sequence of left and right's. As an example "left" once gives us the formula-node $\forall x\,(A \wedge B)$, "left-left-right" gives us B.
If X is a formula-node of Y, and Y is a formula-node of Z, then X is a formula-node of Z. The formula-node X is uniquely identified within Z by combining both 'left/right' sequences.

In a sequent calculus proof, every ancestor of a formula C corresponds to exactly one formula-node within C. In this we disregard substitution for the case of $\forall$ or $\exists$ inferences.

Definition 4.2.2 (Weakly-connected)**.** Let C a formula that has been cut in a proof tree Σ. This cut inference will have the following shape

$$\dfrac{\begin{matrix} Q & R \\ \Gamma \to \Delta, C & C, \Gamma \to \Delta \end{matrix}}{\Gamma \to \Delta.}$$

for proof trees Q and R. We define C to be weakly-introduced if there exists a formula-node A of C such that the following holds:

i) In the proof tree, A is only weakly introduced, in either Q or in R.

ii) For every formula-node D of C: If A is a formula-node of D then D may not occur in an initial sequent in either Q or R.

Definition 4.2.3 (Mirrored)**.** Let Π_1 and Π_2 two proof trees, we call these proof trees *mirrored in all ancestors of* C if and only if for every formula node D of C: The proof trees Π_1 and Π_2 are *mirrored* in D, this is the case if either of the following holds

i) The formula-node D only ever occurs in the antecendent of sequents in Π_1 and the succedent of sequents in Π_2.

ii) The formula-node D only ever occurs in the succedent of sequents in Π_1 and in the antecendent of sequents in Π_2.

Lemma 4.2.4 (Mirrored)**.** *Let P a proof tree with end-sequent $\Gamma \to \Delta$ with as its final inference a cut-inference*

$$\dfrac{\begin{matrix} Q & \qquad & R \\ \Gamma \to \Delta, C & & C, \Gamma \to \Delta \end{matrix}}{\Gamma \to \Delta,}$$

then Q and R are mirrored in all ancestors of C.

Proof. It is easily seen that Q and R are mirrored for all direct ancestors of C since those are all in the succedent of Q and the antecendent of R. Looking at the different inference rules, we find that if the principal formulas are mirrored, then the auxiliary formulas are mirrored as well. Using induction we can therefore conclude that Q and R are mirrored in all ancestors of C. □

Definition 4.2.5 (Markings)**.** Let P a proof tree with as final inference a cut of a formula C. This cut-inference has the following shape

$$\dfrac{\begin{matrix} Q & \qquad & R \\ \Gamma \to \Delta, C & & C, \Gamma \to \Delta \end{matrix}}{\Gamma \to \Delta,}$$

We can construct a *Highlighted* formula-tree of C. This is a formula-tree as specified in (4.2.1) where any number of formula-nodes can be *marked*. A marking gives one of two values to a formula-node: left or right. The meaning of a marking is specifying that a certain formula-node will be removed in the corresponding proof tree. If D is marked left then it will be removed in Q, if it is marked right then it will be removed in R. A formula-node can only be marked once.

A highlighted formula-tree of C is constructed as follows: If a formula-node D is only ever weakly-introduced in Q we mark it *left*, every formula that is only ever weakly-introduced in R and has not been marked, will be marked *right*. For every formula-node E that occurs in an initial sequent, we remove all markings for formula-nodes of E, unequal to E. These markings are removed as we cannot change initial sequents. We perform the same transformation for all marked nodes E.
This highlighted formula-tree will be used as a sort of map in the coming proof.

Lemma 4.2.6 (Cut-Reduction)**.** *For any proof tree P of an end-sequent $\Gamma \to \Lambda$ with as final inference a cut of a weakly-connected formula C. We can construct a proof tree P' such that either*

- *The proof tree P' has the same amount of cuts as P, but its final cut-inference is instead of a reduced formula $\lfloor C \rfloor$ which is not weakly-introduced.*
- *The proof tree P' has strictly less cuts that P*

In both of these cases, the size of P' is strictly less than the size of P where we define size as the amount of strong inferences in the proof tree.

Proof. We prove this lemma by performing induction on the size of Q and R. Let H a highlighting of the formula-tree of C. Our induction hypothesis is that we can transform any sub-tree K of Q or R with end-sequent $\Gamma \to \Delta$ to a proof tree K' with end-sequent $\Gamma' \to \Delta'$, such that every formula D, which occurs in the sequent $\Gamma \to \Delta$ has been replaced with a formula D' where we define D' as follows

- If D is not an ancestor of C, we define $D' := D$.
- If D is marked, and the sequent $\Gamma \to \Delta$ occurs on the same side of this marking, then $D' := \emptyset$. Or in other words: We removed D from the sequent.
- If D is marked on the other side or not marked, we define $D' : \overline{D}$. The formula $\overline{D}$ is defined inductively later in the proof.

Furthermore, if D is mirrored in Q and R then D' is so as well.

Let K end with an inference with principal formula D, we will assume D is an ancestor of C as the transformation is trivial otherwise. We first handle the cases where D has outer connective $\wedge, \vee$ or $\supset$.

Every inference of D falls in one of two situations

$$\text{a:}\quad \frac{\begin{matrix}\Sigma\\ \Gamma \to \Delta, \{A\}, \{B\}\end{matrix}}{\Gamma \to \Delta, \{D\}} \qquad \text{b:}\quad \frac{\begin{matrix}\Pi_1 & \Pi_2\\ \Gamma \to \Delta, \{A\} & \Gamma \to \Delta, \{B\}\end{matrix}}{\Gamma \to \Delta, \{D\}}$$

where we used $\{D\}$ to signify D can either occur in the antecendent or succedent.
Whether we are in situation a) or b) is dependent on whether D occurs in the antecendent or succedent of the lower sequent. Because of the mirrored property of Q and R this means that either every inference in Q is situation a or every inference of D in Q falls in situation b, with R falling in the opposite situation. As such, we will name each side a or side b accordingly.

Per induction assumption we can apply our transformation to all immediate sub-trees of K. This will result in the following proof tree

$$\text{a:}\quad \frac{\begin{matrix}\Sigma'\\ \Gamma' \to \Delta', \{A'\}, \{B'\}\end{matrix}}{\Gamma \to \Delta, \{D\}} \qquad \text{b:}\quad \frac{\begin{matrix}\Pi'_1 & \Pi'_2\\ \Gamma' \to \Delta', \{A'\} & \Gamma' \to \Delta', \{B'\}\end{matrix}}{\Gamma \to \Delta, \{D'\}}$$

We will consider both sides together in order to draw out the mirrored property and to make sure our definitions are consistent. Note that Γ and Δ may possibly not be equal between different inferences on the a or b side. Only A' and B' are equal if they occur on the same side.
We will now split by cases on whether A' and B' occur in the antecendent or succedent on side a)

a): Both A' and B' occur in the antecendent, i.e. Σ' has end-sequent $A', B', \Gamma' \to \Delta'$. We will separate by case on the markings of A and B and specify a definition of $\overline{D}$ and the end-sequent of K'. It is often trivial to infer this end-sequent from Σ', or Π'_1 and Π'_1.

i) If either A or B is marked on the b side, we will define $\overline{D} := \overline{A} \wedge \overline{B}$ and mark D on b side. Finally we infer
$a : \overline{A} \wedge \overline{B}, \Gamma' \to \Delta'; b : \Gamma' \to \Delta'$.

ii) If both A and B are marked on the a side, we define $\overline{D} := \overline{A} \wedge \overline{B}$ and mark it on the a side.
$a : \Gamma' \to \Delta'; b : \Gamma' \to \Delta', \overline{A} \wedge \overline{B}$.

iii) If A is marked on the a side and B is not marked we define $\overline{D} := \overline{B}$ and do not mark D $a : \overline{B}, \Gamma' \to \Delta'; b : \Gamma' \to \Delta', \overline{B}$.

iv) The case where B is marked a side and A is not marked is dual to the previous case with $\overline{D} := \overline{A}$.

v) If neither A nor B is marked we define $\overline{D} := \overline{A} \wedge \overline{B}$, we do not mark D and infer the following:
$a : \overline{A} \wedge \overline{B}, \Gamma' \to \Delta'; b : \Gamma' \to \Delta', \overline{A} \wedge \overline{B}$.

b): The case where A' and B' occur in the succedent is dual to case a).

c): The case where A' occurs in the antecendent and B' in the succedent, i.e. Σ' has end-sequent $A', \Gamma' \to \Delta', B'$ is mostly dual to case a). But we will still consider a few cases for different markings.

i-ii) The cases where either formula is marked b side or both are marked a side are dual to these cases in a).

iii) If A is marked a and B is marked none, we define $\overline{D} := \overline{B}$ and do not mark D furthermore we infer
$a : \Gamma' \to \Delta', \overline{B}; b : \overline{B}, \Gamma' \to \Delta'$.

iv) If B is marked a and A is marked none, we define $\overline{D} := \overline{A}$ and do not mark D. Finally we infer $a : \overline{A}, \Gamma' \to \Delta'$; $b : \Gamma' \to \Delta', \overline{A}$

v) The case where neither A nor B is marked is dual to case v) in a).

This case also demonstrates why we do not simply prove by cases on the outer connective of D, during this transformation formulas may flip sides within a sequent. This may specifically happen in case iv) above. This will change what inferences below it are possible. These changes are consistent across the proof tree as they are dependent on the markings and transformations of their ancestors.

c.2) The case where B' occurs in the antecendent and A' in the succedent, i.e Σ' has end-sequent $B', \Gamma' \to \Delta', A'$ is completely dual to case c).

We will now consider unary inferences

d) Let D have shape $(\forall x)F(x)$ and define A as the auxiliary formula of the final inference in K. If A is marked on the same side as K occurs, then D is removed.
If A is not marked on this side and $\overline{A}$ is on the same side of the sequent as A we perform the same $\forall$ inference to reach the desired end-sequent with $\overline{D} := (\forall x)\overline{A}[x/a]$.
If on the other hand A and $\overline{A}$ are on different sides of the sequent we instead perform an $\exists$ inference and choose $\overline{D} := (\exists x)\overline{A}[x/a]$.

e) The case where D has shape $(\exists x)F(x)$ is dual to case d).

This change from $\forall$ to $\exists$ is necessary as $\forall$ inference might be invalidated when A' changes location from succedent to antecendent.

Finally we will consider cases where D is not the result of a strong inference

f) If D occurs in an initial sequent, then per definition (4.2.5) no formula-nodes of D are marked, therefore $\overline{D} = D$ and as D cannot be marked on the same side as the initial sequent we do not have to change anything.

g) If D is the principal formula of a weakening then we separate on two cases: Either D is marked on the same side as the weakening, in which case we remove it. If D is not marked this way, we change our weakening to instead infer $\overline{D}$ on the correct side of the sequent. Where $\overline{D}$ is defined in the cases above.

Per induction we can apply this transformation to both Q and R. If C is marked then either Q or R will have end-sequent $\Gamma \to \Delta$ where all ancestors of C have been removed. This will be our proof tree Σ'. If on the other hand C is not marked, we can instead make the following inference.

$$\cdot\ \dfrac{\begin{matrix} Q' & R' \\ \Gamma \to \Delta, \overline{C} & \overline{C}, \Gamma \to \Delta \end{matrix}}{\Gamma \to \Delta}$$

Note that since Γ and Δ contain no ancestors of C they have not been changed. Because of our transformation $\overline{C}$ will no longer be weakly-connected and with this we have our desired result. □

Theorem 4.2.7 (Weakly-Connected cut elimination)**.** *Let P be any proof tree of a sequent $\Gamma \to \Delta$, which contains at least one weakly-connected cut. This proof can be transformed to a proof tree P' which contains no weakly-connected cuts and is strictly smaller than P.*

Proof. Since P contains a weakly-connected cut, we can apply the transformation from (4.2.6) on a sub-tree of P to create a new proof tree Σ. This proof tree may have more weakly-connected cuts, but its size will be strictly smaller than P. We can therefore use induction and repeatedly apply this lemma, as the length of a proof tree is finite this process will eventually halt and we will have a proof tree P' which contains no weakly-connected cuts. □

4.3 Weak Cut Removal

Remark 4.3.1. We will now present an alternative, more simple correction. But where the previous solution only decreased the size of our proof tree, this solution instead increases it. Furthermore it is more directly involved with the induction process of the original proof. I believe this proof is rather simple however and merely adds another case to (4.1.9) demonstrating how to remove weakly-introduced cuts without unanchoring formulas.

Lemma 4.3.2 (Weak-cut removal)**.** *Let Σ a proof tree with end-sequent $\Gamma \to \Lambda$ such that the final inference of Σ is a weakly-introduced cut inference*

$$\dfrac{\begin{matrix}Q \\ \Gamma \to \Lambda, C\end{matrix} \qquad \begin{matrix}R \\ \Gamma \to \Lambda, C\end{matrix}}{\Gamma \to \Lambda}$$

Where C has depth d and all other free-cuts in Σ have a depth less than d.

*We can now transform Σ to a proof tree Σ' with the same end-sequent, such that Σ' contains only free-cuts with depth less than d. Furthermore, all formulas in $\Gamma \to \Delta$ which were anchored or **not** weakly introduced, still have these properties in Σ'.*

Proof. We separate by case on the shape of C.

a): The formula C has shape $A \wedge B$. If C is weakly introduced in R then we can remove all ancestors of C in R to get an inference of $\Gamma \to \Delta$, we then construct the following proof tree:

$$\dfrac{\dfrac{\dfrac{\dfrac{R'}{\Gamma \to \Lambda}}{B, \Gamma \to \Lambda}}{A, B, \Gamma \to \Lambda}}{A \wedge B, \Gamma \to \Lambda}$$

The formula C i.e. $A \wedge B$ is now no-longer weakly-introduced. If C is weakly-introduced in Q we make a similar transformation as with R except the final inference differs

$$\dfrac{\dfrac{\dfrac{Q'}{\Gamma \to \Lambda}}{\Gamma \to \Lambda, A} \qquad \dfrac{\dfrac{Q'}{\Gamma \to \Lambda}}{\Gamma \to \Lambda, B}}{\Gamma \to \Lambda, A \wedge B}$$

After these transformation(s) our cut will no longer be weakly-introduced and we can therefore apply (4.1.9).

b-c): The case where C has shape $A \vee B$ or $A \supset B$ are dual to a).

d): If C has shape $(\forall x)F(x)$ we make a similar transformation as with case a) except our transformations of Q and R will look slightly different

$$\dfrac{\dfrac{\dfrac{Q'}{\Gamma \to \Lambda}}{\Gamma \to \Lambda, F(a)}}{\Gamma \to \Lambda, (\forall x)F(x)}$$

where a is any free-variable not yet used in Σ. Similarly we transform R:

$$\dfrac{\dfrac{\dfrac{R'}{\Gamma \to \Lambda}}{F(t), \Gamma \to \Lambda}}{(\forall x)F(x), \Gamma \to \Lambda}$$

Where t is any term.

e): The case where C has shape $(\exists x)F(x)$ is dual to case d.

f): If C is atomic we mostly copy case f) from (4.1.9). This does not need to be changed if C is only weakly-introduced in R. The case where C is weakly-introduced in Q is dual to this.

If C is only weakly-introduced in both Q and R then we make the following change to our induction step. We remove all ancestors of C in R to create a proof R' of $\Gamma \to \Lambda$.
In the transformation of Q in (4.1.9) every initial sequent $\Pi \to \Delta$ was transformed to $\Pi, \Gamma \to \Lambda, \Delta$ (unless an ancestor of C occurred in Δ) this sequent could be easily inferred from $\Pi \to \Delta$.
As C is weakly-introduced in Q there are no initial sequents containing ancestors of C. We still wish to include the proof tree R in our result however. As such we pick a random sequent $\Pi, \Gamma \to \Lambda, \Delta$ and instead of inferring it in the method described above, we copy the proof tree R on top of it and infer the sequent $\Pi, \Gamma \to \Lambda, \Delta$ from $\Gamma \to \Lambda$ using weak-inferences.

□

Remark 4.3.3. With this we have demonstrated a second alternative method to solve the error in (4.1.9). This new method is simpler, but also performs worse in terms of tree-size. The correction as shown in [2] is formulated as a correction on the upper bound of tree-size after removing free-cuts. This method changes the theorem and makes it stronger.

Next to these considerations on tree-size and strength of the theorem, I most of all consider the *reduction* an interesting transformation which can possibly be applied further to remove unnecessary parts of proof trees.

5 Normal Form

5.1 Weak Normal Form

In order to more easily compare Natural Deduction to Sequent calculus, we need to apply more restrictions to our system. This means we can more easily predict the shape of proof trees. In this section we will introduce a weak normal form for proof trees. This restricts us in naming of free variables and also disallows some 'useless' inferences.

Everything in this chapter is rewritten theory from Prawitz 'Natural Deduction' [3].

Definition 5.1.1 (Free variable normal form)**.** A proof tree Σ is in free variable normal form if and only if every free variable a only occurs as the eigenvariable in a maximum of one $\forall I$ or $\exists E$ inference. Furthermore if a occurs as eigenvariable in such an inference. Then a may only occur in the sub-tree above this inference.

Lemma 5.1.2 (Free variable normal form)**.** *Any proof tree Σ of $\Gamma \vdash A$ can be transformed to a proof tree Σ' of $\Gamma \vdash A$ which is in free variable normal form.*

Proof. We can easily construct such a proof tree by only renaming variables. Let Σ a proof tree of $\Gamma \vdash A$. We can find an infinite set K of free variables not used in Σ. For every $\forall I$ or $\exists E$ inference we rename the eigenvariable a to the next variable b in K. This will not invalidate any inferences outside this sub-tree. As all assumptions containing a have to be discharged in order to perform $\forall I$ or $\exists E$. □

Definition 5.1.3 (Major/Minor assumptions)**.** Some inference rules are less symmetrical than others, for these we differentiate between *Major* and *Minor* assumptions. The minor assumption is often the assumption where we do not know its outer connective.

$\vee E$ We define $A \vee B$ as the major assumption while both occurrences of C are minor assumptions.

$\supset E$ We define $A \supset B$ as the major assumption and A as the minor assumption.

$\exists E$ We define $\exists x\, Fx$ as the major assumption and C as the minor assumption.

All other assumptions are major assumptions.

Definition 5.1.4 (Weak Normal Form)**.** A formula occurrence is a *maximal formula* if and only if it is both the result of an introduction rule and the major assumption of an elimination rule.
A proof tree Σ is in *Weak Normal Form* if and only if Σ is in free variable normal form and Σ does not contain any maximal formulas.

Lemma 5.1.5 (Weak Normal Form)**.** *Any proof tree P of $\Gamma \vdash A$ can be transformed to a proof tree P' of $\Gamma \vdash A$ which is in Weak Normal Form.*

Proof. We will demonstrate how to remove every maximal formula D from our proof tree. If a formula D is the result of an inference rule, it will have the corresponding outer connective. There is then only one elimination for which the major assumption has the same outer connective. This allows us to split our proof into one case for each different connective.

a): If D has outer connective $\wedge$, we will transform

$$\dfrac{\dfrac{\dfrac{\Sigma_1}{A} \quad \dfrac{\Sigma_2}{B}}{A \wedge B}}{\dfrac{A}{\Pi}} \qquad \text{to:} \qquad \dfrac{\dfrac{\Sigma_1}{A}}{\Pi}$$

The case where $\wedge E$ has conclusion formula B is dual to this case.

b): If D has outer connective $\vee$, we will transform

$$\dfrac{\dfrac{\Sigma_1}{A}}{\dfrac{A \vee B}{\dfrac{C}{\Pi}}} \quad \dfrac{\dfrac{[A]}{\Sigma_2}}{C} \quad \dfrac{\dfrac{[A]}{\Sigma_2}}{C} \qquad \text{to:} \qquad \dfrac{\dfrac{\Sigma_1}{[A]}}{\dfrac{\Sigma_2}{\dfrac{C}{\Pi}}}$$

Where we copied the proof tree Σ_1 onto every assumption of shape $[A]$. The case where Σ_1 has conclusion B is dual to this.

c): If D has outer connective $\supset$, we will transform

$$\dfrac{\dfrac{\Sigma_1}{A} \quad \dfrac{\dfrac{[A]}{\Sigma_2}}{B}}{\dfrac{A \supset B}{\dfrac{B}{\Pi}}} \qquad \text{to:} \qquad \dfrac{\dfrac{\Sigma_1}{[A]}}{\dfrac{\Sigma_2}{\dfrac{B}{\Pi}}}$$

d): If D has outer connective $\exists$, we first rename variables in order to ensure our proof tree is in free variable normal form. Then we transform

$$\dfrac{\dfrac{\dfrac{\Sigma_1}{Fa}}{\exists x\, Fx} \quad \dfrac{\dfrac{[Fb]}{\Sigma_2}}{C}}{\dfrac{C}{\Pi}} \qquad \text{to:} \qquad \dfrac{\dfrac{\Sigma_1[b/a]}{[Fb]}}{\dfrac{\Sigma_2}{\dfrac{C}{\Pi}}}$$

In this, $\Sigma_1[b/a]$ signifies that we renamed every occurrence of variable b to a in the proof tree. This renaming does not cause any issues, as our proof tree is in free variable normal form and as such Σ_1 does not contain any occurrences of a which might conflict with this renaming.

e): If D has outer connective $\forall$ we will transform

$$\dfrac{\dfrac{\dfrac{\Sigma}{Fa}}{\forall x\, Fx}}{\dfrac{Fb}{\Pi}} \qquad \text{to:} \qquad \dfrac{\Sigma[b/a]}{\dfrac{Fb}{\Pi}}$$

When removing a maximal formula it might cause the assumption formula above D to become a maximal formula. As an example in the proof tree in case *a*: If Σ_1 ends with an introduction rule and Π 'starts' with an elimination rule, then after our transformation A will instead be a maximal formula.
However, each transformation decreases the size of our proof tree P therefore per induction we can remove all maximal formulas from P. At the end of this transformation we apply (5.1.2) to find our desired P' in weak normal form. □

5.2 Normal Form

In order to define a normal form for our natural deduction proofs we need to place more restrictions on our system. These restrictions will involve removing the symbols $\exists$ and $\vee$ from our system. With this change, we can no longer construct proofs for all formulas, but we can instead prove equivalent ones.

Remark 5.2.1. We will first justify that we can remove the symbols $\exists$ and $\vee$. We will do this by showing that can instead prove equivalent statements which do not contain these symbols.

Definition 5.2.2 (Weakly Definable)**.** A connective γ is *weakly definable* in a system S if for every formula A there exists a transformation A^* not containing γ such that

i) The formula A^* is obtained by replacing all sub-formulas of the shape $B\gamma C$ with $(B\gamma C)^*$, or if γ is a unary connective then $\gamma x\, B$ is replaced with $(\gamma\, xB)$.

ii) We have $\vdash A^*$ if and only if $\vdash A$

Lemma 5.2.3. *The symbols $\lor$ and $\exists$ are weakly definable in the system of natural deduction.*

Proof. We will first demonstrate that the connective $\lor$ is strongly definable. Every formula A can be transformed to a formula A^* which does not contain $\lor$. We do this by recursively replacing sub-formulas of the shape $B \lor C$ with $\neg B \supset C$.

We will demonstrate how to perform $\lor I$ and $\lor E$ using this new definition on formulas A^*, from this it will follow that any proof tree of A can be transformed to a proof tree of A^* and vice versa.

$$\dfrac{\dfrac{\Sigma}{A}}{A \lor B} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\Sigma}{A} \quad \neg A}{\bot}}{B}}{\neg A \supset B}$$

Or if we infer $A \lor B$ from B:

$$\dfrac{\dfrac{\Sigma}{B}}{A \lor B} \qquad \dfrac{\dfrac{\Sigma}{B}}{\neg A \supset B}$$

And finally we have $\lor E$

$$\dfrac{\dfrac{\Sigma_1}{A \lor B} \quad \dfrac{\dfrac{[A]}{\Sigma_2}}{C} \quad \dfrac{\dfrac{[B]}{\Sigma_3}}{C}}{C} \qquad \dfrac{\dfrac{C^2 \quad \dfrac{\dfrac{\dfrac{\dfrac{C^2 \quad \dfrac{\dfrac{\dfrac{\neg A^1 \quad \dfrac{\Sigma_1}{\neg A \supset B}}{[B]}}{\Sigma_3}}{C}}{\bot}}{[A]}\,1}{\Sigma_2}}{C}}{\bot}}{C}\,2$$

Now we will demonstrate that $\exists$ is strongly definable. Every formula A can be transformed to a formula which does not contain $\exists$. We do this by recursively replacing sub-formulas of the shape $\exists x\, B$ with $\neg(\forall x\, \neg B)$. We can replace $\exists I$ with

$$\dfrac{\dfrac{\Sigma}{Fa}}{\exists x\, Fx} \qquad \dfrac{\dfrac{\dfrac{\Sigma}{Fa} \quad \dfrac{\forall x\, \neg Fx}{\neg Fa}}{\bot}}{\neg \forall x\, \neg Fx}$$

We can replace $\exists E$ with

$$\dfrac{\dfrac{\Sigma_1}{\exists x\, Fx} \quad \dfrac{\dfrac{[Fa]}{\Sigma_2}}{C}}{C} \qquad \dfrac{\dfrac{\dfrac{\Sigma_1}{\neg \forall x\, \neg Fx} \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[Fa]}{\Sigma_2}}{C} \quad \neg C^1}{\bot}}{\neg Fa}}{\forall x\, \neg Fx}}{\bot}}{C}\,1$$

As desired. $\square$

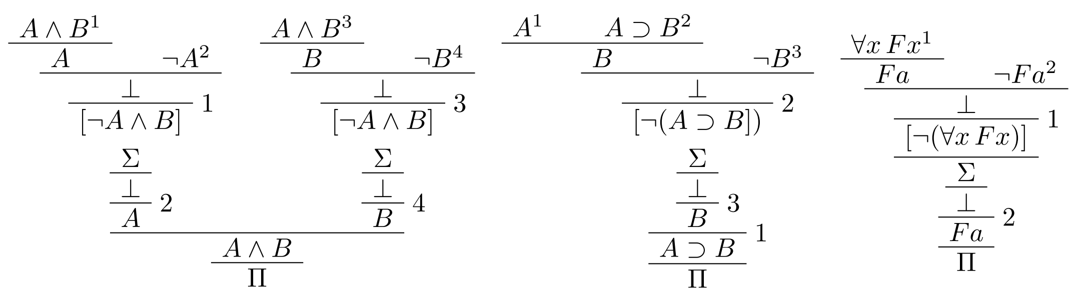
Definition 5.2.4 (Normal Form)**.** A proof tree P is in normal form, if and only if it is in weak-normal form and all $\bot_c$ inferences in P are atomic. Namely for every $\bot_c$ inference in P the conclusion formula is an atomic formula.

Theorem 5.2.5 (Normal Form)**.** *Any proof tree P of $\Gamma \vdash A$ can be transformed to a proof tree P' in normal form of $\Gamma \vdash A$.*

Proof. We will prove this using induction, our induction value will be a tuple $\langle d, k\rangle$. Where d is the maximum depth of $\bot_c$-inferences in our proof tree P and k is the amount of $\bot_c$ of this depth. We will say that $\langle d_1, k_1\rangle < \langle d_2, k_2\rangle$ if either $d_1 < d_2$ or both $d_1 = d_2$ and $k_1 < k_2$.
We will demonstrate how to transform a $\bot_c$-inference of depth d for d larger than 0 to a $\bot_c$-inference of depth $d-1$. This transformation will only affect that $\bot_c$-application and will therefore decrease our induction value. Such a $\bot_c$-inference will look as follows:

$$\begin{array}{c}[\neg D]\\ \hline \Sigma\\ \hline \bot\\ \hline D\\ \hline \Pi\end{array}$$

We will separate cases on the shape of D, since D is not atomic it can either have shape $A \wedge B, A \supset B$ or $\forall x\, Fx$. From left to right we have shown the corresponding transformed proof tree:



Because the value $\langle d, k\rangle$ can only be decreased a finite amount of times[3], it follows per induction that we can transform any proof tree P to a proof tree P' in which all $\bot_c$-inferences are atomic. We can then apply (5.1.5) to transform this to a proof tree P'' which is also in weak-normal form. As this transformation does not influence $\bot_c$ applications the proof tree P'' will have all desired properties. □

5.3 Properties

After removing $\vee$ and $\exists$ symbols we are left with an interesting property. Of all inferences, only $\supset E$ has a minor formula. This is also the only introduction rule where an upper formula is not a sub-formula of the lower formula. Everything else follows a more predictable pattern.

Definition 5.3.1 (branches)**.** A branch π of a proof tree P is a series of formula occurrences $A_1, ..., A_n$ in P, such that

i) The formula occurrence A_1 is either the end-formula or a minor premise of an $\supset E$-inference.

ii) The formula occurrence A_n is a leaf in P.

iii) For every integer $1 \le i < n$ the inference of formula A_i has the formula A_{i+1} as a major premise.

As an example the following proof tree has three branches

[3] Only one of the cases increases the amount of $\bot_c$ inferences. Therefore when decreasing our value d by one, the amount of $\bot_c$ inferences is at most doubled. Therefore our induction would take a maximum of $n \cdot 2^d$ steps, where n is the total amount of $\bot_c$-inferences in P

$$\dfrac{\dfrac{\dfrac{\dfrac{A^2 \qquad A \supset B^3}{B}}{A^1}}{A \wedge B}\supset I,3}{(A \supset B) \supset A \wedge B}\supset I,1,2$$
$$A \supset (A \supset B) \supset A \wedge B$$

Specifically: one branch contains the sequence from A^1 to the end-formula. The second branch contains only A^2 and the third branch is the sequence from $A \supset B^3$ to the end-formula.

Definition 5.3.2. We define an *order* on branches within a proof tree P. We say that any branch $\pi = A_1, ..., A_n$ such that A_n is the end-formula of P has order 0 and is defined as a *main branch.*
If on the other hand A_n is the minor premise of an $\supset E$-inference and the conclusion formula belongs to a branch of order n then we say π has order $n + 1$.

Lemma 5.3.3. *Let P be a proof tree of $\Gamma \vdash A$ in normal form and let π be any branch $A_1, ..., A_n$ in P. There now exists an integer i such that A_i is the minimal formula of this branch. This formula has the following properties:*

i) Every formula A_j for $j < i$ is a major premise of an E-rule and contains A_{j+1} as a sub-formula.

ii) The formula occurrence A_i is either the major premise of an I-rule or the major premise of an $\perp_c$-inference.

iii) Every formula A_k for $i < k < n$ is the major premise of an I-rule and a sub-formula of A_{k+1}.

Proof. As our proof P is in normal form, it follows that no formula in π is a maximal formula. If exists a formula A_i which is the major premise of an I-rule such that A_{i+1} is the major premise of an E-rule, then it follows that A_{i+1} is a maximal formula. From this we can see that all E-inferences must occur before I-inferences. All that is left is to consider the $\perp_c$-rule.

Assume a formula A_i in π is the major premise of an $\perp_c$-inference. The formula A_i is of the shape $\perp$ and is therefore not the result of an I-inference, it is either a leaf or the result of an E-inference. Since every $\perp - c$ application is atomic, it follows that the conclusion formula A_{i+1} is atomic as well. It therefore cannot be the major premise of an E-rule and because it cannot have shape $\perp$ we either have $i + 1 = n$ or A_{i+1} is the major premise of an I-rule. □

We can view this as every branch having an elimination part and an introduction part.

Remark 5.3.4. We will not prove it in this thesis, but an interesting result from this normal form is the sub-formula property. Every formula in a normal form proof tree is either

- A sub-formula of the end-formula,
- A sub-formula of an open assumption,
- or discharged by a $\perp_c$ inference and has the atomic formula $\perp$ directly below it in the proof-tree.

6 Equivalence between Normal Forms

6.1 Natural Deduction models Sequent calculus

We will work in C' for this chapter, as we will work with normal forms. Though the proof in this first section could easily be expanded to include cases for $\vee$ and $\exists$ as can be seen in [6] and [3].

Lemma 6.1.1. *For every sequent calculus proof P of $\vdash A_1, ..., A_j \to B_k, ..., B_1$ there exists a natural deduction proof P' of $A_1, ..., A_j, \neg B_2, ..., \neg B_k \vdash B_1$, where we interpret the empty succedent as containing $\bot$.*

Proof. We will prove this using induction on the height of P. We will split by cases on the final inference of P, per induction assumption the lemma is true for all immediate sub-trees Σ_i of P.

If P is an assumption tree which only contains $A \to A$ or $\bot \to$ then the natural deduction proof tree is trivial, namely the assumption trees A and $\bot$.

$$\dfrac{\dfrac{\Sigma_1}{\Gamma \to \Delta, A} \quad \dfrac{\Sigma_2}{\Gamma \to \Delta, B}}{\Gamma \to \Delta, A \wedge B} \qquad \dfrac{\dfrac{\Sigma_1'}{A} \quad \dfrac{\Sigma_2'}{B}}{A \wedge B}$$

$$\dfrac{\dfrac{\Sigma_1}{A, B, \Gamma \to \Delta}}{A \wedge B, \Gamma \to \Delta} \qquad \dfrac{\dfrac{A \wedge B}{[A]} \quad \dfrac{A \wedge B}{[B]}}{\Sigma_1'}$$

$$\dfrac{\dfrac{\Sigma_1}{A, \Gamma \to \Delta, B}}{\Gamma \to \Delta, A \supset B} \qquad \dfrac{\dfrac{\dfrac{[A]}{\Sigma_1'}}{B}}{A \supset B}$$

$$\dfrac{\dfrac{\Sigma_1}{\Gamma \to \Delta, A} \quad \dfrac{\Sigma_2}{B, \Gamma \to \Delta}}{A \supset B, \Gamma \to \Delta} \qquad \dfrac{\dfrac{\Sigma_1'}{A} \quad A \supset B}{\dfrac{[B]}{\Sigma_2'}}$$

$$\dfrac{\dfrac{\Sigma_1}{\Gamma \to \Delta, Fa}}{\Gamma \to \Delta, \forall x\, Fx} \qquad \dfrac{\dfrac{\Sigma_1'}{Fa}}{\forall x\, Fx}$$

$$\dfrac{\dfrac{\Sigma_1}{Ft, \Gamma \to \Delta}}{\forall x\, Fx, \Gamma \to \Delta} \qquad \dfrac{\dfrac{\forall x\, Fx}{[Ft]}}{\Sigma_1'}$$

$$\dfrac{\dfrac{\Sigma_1}{\Gamma \to \Delta, C} \quad \dfrac{\Sigma_2}{C, \Gamma \to \Delta}}{\Gamma \to \Delta} \qquad \dfrac{\dfrac{\Sigma_1'}{[C]}}{\Sigma_2'}$$

$$\dfrac{\dfrac{\Sigma_1}{\Gamma \to \Delta, A, B}}{\Gamma \to \Delta, B, A} \qquad \dfrac{\dfrac{\dfrac{[\neg A]^1}{\Sigma_1'}}{B} \quad \neg B}{\dfrac{\bot}{A}\, \bot_c, 1}$$

We only examine the exchange involving the rightmost formula in the succedent. All other exchanges are trivial as natural deduction does not keep track of the order of assumptions. Similarly we will not consider weakenings and contractions.

Without loss of generality we can assume that P is in free variable normal form, therefore for the case $\forall : right$, we will not generate any new assumptions containing a. □

6.2 Normal Sequent Calculus models Normal Natural deduction

We will show that any normal natural deduction proof can be transformed to a normal sequent calculus proof. This new normal form will be cut-free but with some additional restrictions. This proof is my own but was inspired by the proof that sequent calculus models natural deduction from [3]. Similarly this definition of the sequent calculus normal form is my own and I have not been able to find it in my research, though I would not be surprised if it already exists.

Definition 6.2.1. A normal form sequent calculus proof, is cut-free and for every sequent $A_1, ..., A_j \to B_k, ..., B_1$ in the proof tree, the formulas $B_k, ..., B_2$ are all atomic.

Remark 6.2.2. This change is equivalent to the normal form property that all $\perp_c$ inferences must be atomic. This can be seen in the previous section where exchanges in the succedent had to be rewritten as $\perp_c$ inferences.

Definition 6.2.3. We define the *maximum order* of a proof tree Σ in natural deduction to be the maximum of the order of all branches in Σ.

Theorem 6.2.4. *Let Γ be a set of formulas and Δ any set of atomic formulas, such that for every $B \in \Delta$ there exists an $(\neg B) \in \Gamma$.*

For any normal-form proof tree P in natural deduction of $\Gamma \vdash A$, there exists a normal-form proof tree P' in sequent calculus of

- *the sequent $\Gamma^- \to \Delta$ if A has the shape $\perp$.*
- *the sequent $\Gamma^- \to \Delta, A$ otherwise.*

Where $\Gamma^- = \Gamma \setminus \{\neg B | B \in \Delta\}$.

Remark 6.2.5. We will prove this theorem by induction on the maximum order of proof P. To do this we will show that we can specifically replicate branches in natural deduction to proofs in sequent calculus.

We have formulated the following two lemmas demonstrating how to replicate both the E-part and the I-part respectively of a branch. Each of these lemmas is formulated with induction in mind.

Remark 6.2.6. The goal of this proof was to transform a natural deduction proof into a sequent calculus proof without introducing cuts. We would naturally prove this using induction on each of the inferences in these proofs. The first issue is that we cannot perform elimination inferences in sequent calculus.

We instead use the properties of normal form natural deduction proofs we discovered in the previous chapters. Every branch starts with a series of elimination inferences the 'E-part' of our branch, then we reach a middle formula which is followed by a series of introduction rules. If we now start with the middle formula and then simulate elimination rules as inferences in the antecendent. We will get our desired proof tree.

The other issue is $\perp_c$ inferences. The equivalent inference in sequent calculus would move a formula $\neg B$ from the antecendent to the succedent. This can only be done using a cut. The way we solve this is by making sure that B already occurs in the succedent and that we never add $\neg B$ to the antecendent. We use the property that sequent calculus allows form multiple formulas in its succedent. But because our normal form for natural deduction specifies that all $\perp_c$ inferences must be atomic, we can in fact restrict the succedent to only allow for additional atomic formulas and contain just one non-atomic formula.

Lemma 6.2.7. *Let P a normal form natural deduction proof tree with maximum order $n+1$ and let the* (6.2.4) *theorem hold for all proofs with maximum order less than or equal to n.*

If all branches of order 0 in P have an empty I-part. Then the theorem holds for P.

Proof. Firstly we note that P only has one branch π of order 0 as all elimination rules ($\supset E$, $\wedge E$ and $\forall E$) have only one major assumption. We will prove our lemma by induction on the length of this branch.

Case 1) Base case: If π has length one then P contains only the assumption tree A_i. We seperate by case on the shape of A_i.

- If A_i has the shape $\bot$ then we use the axiom $\bot \to$ to infer $\bot \to \Delta$.
- If A_i has the shape $\neg B$ for B in Δ, then we start with axiom $B \to B$ and infer $\to B, \neg B$ from which we can find $\to \Delta, \neg B$ as desired.
- If none of the cases above hold then we use the axiom $A_i \to A_i$ and perform weakenings to infer $A_i \to \Delta, A_i$.

Note that removing $\neg B$ from the antecendent in case 2 might invalidate some inferences in our induction cases. It can easily be added back however.

Case 2) Induction case: We will assume that our lemma is true if π has length k or less. We will show that it holds for a branch of length $k+1$.

Case 2a) If A_1 has shape $A \wedge B$ then the proof-tree P has the following shape

$$\dfrac{\dfrac{A \wedge B}{A}}{\Sigma}$$

The main branch of Σ has length k and therefore per induction assumption we know there exists a proof-tree Σ' of $A, \Pi \to \Delta, A_i$ where A_i is the minimal formula of π and also the end-formula of P and Σ. If A is not contained in the antecendent we add it by weakening. We can now construct our desired proof tree P' as follows

$$\dfrac{\dfrac{\dfrac{\dfrac{\Sigma'}{A, \Pi \to \Delta, A_i}}{A, B, \Pi \to \Delta, A_i}}{A \wedge B, \Pi \to \Delta, A_i}}{}$$

Case 2b) The case where A_1 has shape $\forall x\, Fx$ is very similar to case $2a$. The proof tree P will have shape

$$\dfrac{\dfrac{\forall x\, Fx}{Ft}}{\Sigma}$$

and in sequent calculus we will find Π' as follows

$$\dfrac{\dfrac{\Sigma'}{Ft, \Gamma^- \to \Delta, A_i}}{\forall x\, Fx, \Gamma^- \to \Delta, A_i}$$

Case 2c) For the case where A_1 has shape $A \supset B$, the proof tree P has the following shape

$$\dfrac{\dfrac{\dfrac{Q}{A} \qquad A \supset B}{B}}{\Sigma}$$

The sub-tree Q has maximum order less than or equal to n. The order of every branch in Q is one higher in P as they all branch off our main branch π. As an example the main branch of Q with end-formula A has order 0 in Q but order 1 in P.

Per assumption in our lemma we can create a proof-tree Q' of $\Pi \to \Delta, A$. Per our induction assumption we can also create a proof tree Σ' of $B, \Gamma \to \Delta, A_i$. We can now construct the following proof tree P'

$$\dfrac{\dfrac{Q'}{\Pi \to \Delta, A} \qquad \dfrac{\Sigma'}{B, \Gamma \to \Delta, A_i}}{A \supset B, \Gamma, \Pi \to \Delta, A_i}$$

If B has the shape $\bot$ and the formula A is atomic and contained in Δ, then we will instead use only Q' to infer

$$\dfrac{\dfrac{Q'}{\Pi \to \Delta}}{\Gamma, \Pi \to \Delta}$$

as desired.

Per induction it now follows that our lemma holds for any length of π. Therefore it holds for all P with maximum order $n+1$ and no I-part to their main branch. □

Lemma 6.2.8. *Let P a normal form natural deduction proof tree with maximum order $n+1$ and let the* (6.2.4) *theorem hold for all proofs with maximum order less than or equal to n. Then the theorem also holds for P.*

Proof. We now have a set of main branches, we will perform induction on the maximum length of the I-part in branches in this set.

Case 1) Base Case: If the I-part is empty for all branches in P then the lemma is true per (6.2.7) above.

Case 2 Induction Case: Let the maximum length be equal to $n+1$ and the lemma above be true for all proof trees with maximum length equal to n.

Case 2a) If A_n the end-formula has shape $A \wedge B$ then the proof tree P has shape

$$\dfrac{\dfrac{Q}{A} \qquad \dfrac{R}{B}}{A \wedge B}$$

Per induction assumption there exist proofs Q' and R' of sequents $\Gamma^- \to \Delta, A$ and $\Pi^- \to \Delta, B$ respectively. We now construct our desired proof tree P' as follows

$$\dfrac{\dfrac{Q'}{\Gamma^- \to \Delta, A} \qquad \dfrac{R'}{\Pi^- \to \Delta, B}}{\Gamma^-, \Pi^- \to \Delta, A \wedge B}$$

Case 2b) The case where A_n has shape $\forall x Fx$ is dual to case $2a$. The proof tree P has shape

$$\dfrac{\dfrac{Q}{Fa}}{\forall x\, Fx}$$

and we can construct the proof tree P' as follows

$$\dfrac{\dfrac{Q'}{\Gamma^- \to \Delta, Fa}}{\Gamma^- \to \Delta, \forall x\, Fx}$$

This inference is valid because Γ^- and Δ both only contain (sub)-formulas of open assumptions in P and to perform $\forall I$ these open assumptions may not contain a.

Case 2c) The case where A_n has shape $A \supset B$ is also similar, the proof tree P has shape

$$\dfrac{\dfrac{\dfrac{[A]}{Q}}{B}}{A \supset B}$$

Where we discharge all assumptions of the shape A. Per induction assumption there exists a proof-tree Q' of $A, \Gamma^- \to \Delta, B$.
Note we may need to add A (or B if it has shape $\bot$) by weakening.

We can now construct the proof-tree P' as follows

$$\dfrac{\dfrac{Q'}{A, \Gamma^- \to \Delta, B}}{\Gamma^- \to \Delta, A \supset B}$$

Case 2d) There is also the case where A_n is atomic. This is possible if P has the following shape

$$\dfrac{\dfrac{\dfrac{[\neg A_n]}{Q}}{\bot}}{A_n}$$

If we choose $\Delta' := \Delta \cup \{A_n\}$ then per induction assumption we can find a proof-tree of the sequent $\Gamma^- \to \Delta'$ where $\neg A_n$ may not occur in Γ^- and using only exchanges we find our proof of $\Gamma^- \to \Delta, A_n$.

□

Proof of theorem. We can prove our theorem now using induction on the maximum order of proof-trees.
If P has maximum order 0 then our theorem holds for all proof trees with lower order as there are none. Therefore per (6.2.8) our theorem holds for P.

Our induction step immediately follows from (6.2.8) as well. Therefore per induction it follows that our theorem holds for all normal form proof trees. □

6.3 Conclusion

In this section we will shortly discuss the meaning of our results in this chapter. As I have constructed these results independently, this chapter is my own as well.

In the previous chapter we have constructed a method to *directly* transform normal form natural deduction proof trees to normal form sequent calculus proof trees. Showing in the proof a strong connection between the two normal forms.

This construction also gave us the result that every sequent calculus proof tree can be transformed to a normal form sequent calculus proof tree. We have specifically shown the following three transformations. Sequent calculus to Natural deduction, natural deduction to normal form natural deduction trees and finally ND in normal form to sequent calculus in normal form.

This final transformation can also be easily reversed, at the start of this chapter we demonstrated how to transform sequent calculus proofs to natural deduction proofs and this transformation already preserves the normal form. Inference rules in the antecendent are always added as elimination rules at the leaves of the corresponding natural deduction proof tree. Our proof tree will therefore naturally be in weak normal form. This also gives us an unexpected correspondence between cuts in sequent calculus and maximal formulas in natural deduction.

Finally, we have already seen that exchanges in the succedent correspond to $\bot_c$ inferences in natural deduction, therefore we indeed find that the normal forms of sequent calculus and natural deduction are equivalent.

References

[1] Samuel R. Buss. "An introduction to proof theory. Handbook of proof theory". In: *Bulletin of Symbolic Logic* 137.4 (2000), pp. 1–78. DOI: 10.2307/420968.

[2] Arnold Beckmann and Samuel Buss. "Corrected upper bounds for free-cut elimination". In: *Theor. Comput. Sci.* 412 (Sept. 2011), pp. 5433–5445. DOI: 10.1016/j.tcs.2011.05.053.

[3] Dag Prawitz. *Natural Deduction*. Dover Publications, 2006. ISBN: 0486446557.

[4] Morten Heine Sørensen and Pawel Urzyczyn. *Lectures on the Curry-Howard Isomorphism, Volume 149 (Studies in Logic and the Foundations of Mathematics)*. USA: Elsevier Science Inc., 2006. ISBN: 0444520775.

[5] Gerhard Gentzen. "Investigations into Logical Deduction". In: *American Philosophical Quarterly* 1,2.4 (1964), pp. 288–306. ISSN: 00030481. URL: http://www.jstor.org/stable/20009142 (visited on 03/22/2023).

[6] A.S. Troelstra and H. Schwichtenberg. *Basic Proof Theory*. Vol. 7. 1998. DOI: 10.1023/A:1008226228293.