. . . .

Alexander Leitsch

TECHNISCHE UNIVERSITÄT WIEN
VIENNA UNIVERSITY OF TECHNOLOGY

DIPLOMARBEIT

Projection-Based Cut-Elimination and Normalization

ausgeführt am

Institut für Computersprachen
Arbeitsgruppe Theoretische Informatik und Logik
der Technischen Universität Wien

unter der Anleitung von

Univ.Prof. Dr.phil. Alexander Leitsch

durch

Stefan Hetzl
Bergheidengasse 8/1/9
A-1130 Wien

Wien, 24. März 2004

. . . .

Stefan Hetzl

. . . .

Alexander Leitsch

TECHNISCHE UNIVERSITÄT WIEN
VIENNA UNIVERSITY OF TECHNOLOGY

DIPLOMA THESIS

Projection-Based Cut-Elimination and Normalization

carried out at the

Institute of Computer Languages
Theory and Logic Group
of the Vienna University of Technology

under the instruction of

Univ.Prof. Dr.phil. Alexander Leitsch

by

Stefan Hetzl
Bergheidengasse 8/1/9
A-1130 Wien

Vienna, March 24, 2004

. . . .

Stefan Hetzl

Acknowledgments

I want to thank my advisor, professor Alexander Leitsch for providing a highly interesting topic and for many insightful discussions. I am also grateful for his support going beyond that directly related to this thesis.

I would like to thank my parents, Ernst and Eleonore Hetzl for making my studies possible and for providing all the assistance I needed.

This work is dedicated to Christine for the patience and understanding she can bring up for a student of formal logic.

Contents

Chapter 1

Introduction

In 1934 Gentzen published his work "Untersuchungen über das logische Schließen"[7] where he introduced two calculi that became very important for proof theory: the *sequent calculus* and the calculus of *natural deduction*. The main part of this work consists of the proof of the "Hauptsatz" (or cut-elimination theorem) which states that for every valid formula there exists a proof without "detours". Without detours in this context means that the proof consists *only* of subformulas of the original formula. This has interesting mathematical consequences, e.g. that the concepts present in a proposition are sufficient to prove the proposition, or more technically, that every proof depending on arbitrary lemmas can be transformed into a proof that does not use lemmas at all.

Gentzen's proof of the cut-elimination theorem proceeds by shifting a cut upwards in the proof until this is no longer possible. Then the cut is broken up into one or more cuts with cut formulas that all have a lower logical complexity than the original cut formula. While this is a nice inductive proof it is - if regarded as an algorithm - rather inefficient. With the availability of computers the development of cut elimination *algorithms* became an interesting issue with the aim of not only being able to eliminate all cuts but also in doing so efficiently both in theory and practice. From this line of research pursued by Alexander Leitsch and Matthias Baaz two algorithms evolved: cut elimination and redundancy elimination by resolution [3] and the cut-projection method [1, 2].

This diploma thesis deals with the cut-projection method which differs substantially from Gentzen's method. Instead of shifting the cuts upwards they are left in place but the logical complexity is reduced nonetheless. This is achieved by replacing a subformula F of the cut formula by a subformula of F in the cut and in the proof above it. This transformation decreases the logical complexity of the cut formula without increasing the size of the proof. The price to pay for such a transformation is reduced generality. We can not expect to reduce every cut without increasing the proof size because this would lead to a linear cut-elimination procedure, while, in general, the problem of cut-elimination is of far higher complexity.

The aim of this thesis is first to present the cut-projection method in the sequent calculus (where it was originally developed) but also to show how it can also be applied to proofs in natural deduction. Both of these formalisms will be investigated for intuitionistic logic only. In Chapter 2 the basic notations and definitions are introduced. We start with first-order logic (terms, formulas, ...) and proceed with Gentzen's calculi: the sequent calculus **LJ** and natural deduction **NJ**. We also introduce the lambda calculus in its type-free and typed variants. Chapter 3 is a discussion of the relations between the defined formalism. We will give translations of proofs from the sequent calculus to natural deduction and vice versa and present the Curry-Howard isomorphism, a very close correspondence between natural deduction and the typed lambda calculus. Chapter 4 discusses the reductions in these formalisms: cut-elimination for the sequent calculus and normalization in natural deduction (and the typed lambda calculus). The cut-elimination theorem as well as the important properties of confluence and termination of normalization will be proved. Finally, in Chapter 5 the method of cut projection is introduced for the sequent calculus and it is shown that cuts in a certain syntactic subclass can be eliminated with only exponential expense by using cut projection. The novelty in this thesis consists of an extension of the cut projection method to cover also natural deduction (and the typed lambda calculus) discussed in the second part of Chapter 5.

Chapter 2

Notations and Definitions

2.1 First-Order Logic

Definition 2.1 (Language). The language of first-order logic consists of the following elements:

- A set of *variables* $V = \{x, y, z, x_1, x_2, \ldots\}$
- A set of *constant symbols* $CS = \{a, b, c, a_1, a_2, \ldots\}$
- For every $n \geq 1$ a set of *function symbols* FS_n with arity n. $FS = \bigcup_{n \geq 1} FS_n$ is the set of all function symbols.
- For every $n \geq 1$ a set of *predicate symbols* PS_n with arity n. $PS = \bigcup_{n \geq 1} PS_n$ is the set of all predicate symbols.
- Propositional connectives $\vee$,$\wedge$,$\rightarrow$ and $\neg$
- Quantifiers $\forall$,$\exists$
- $\top$ (verum) and $\bot$ (falsum)

Definition 2.2 (Terms). The set of terms T is defined inductively:

1. $V \subseteq T$
2. $CS \subseteq T$
3. For all $n \geq 1$: If $t_1, \ldots, t_n \in T$ and $f \in FS_n$ then $f(t_1, \ldots, t_n) \in T$

Definition 2.3 (Formulas). The set of first-order formulas PL is defined inductively:

1. For all $n \geq 1$: If $P \in PS_n$ and $t_1, \ldots, t_n \in T$ then $P(t_1, \ldots, t_n) \in \mathrm{PL}$

2. $\top \in \mathrm{PL}$, $\bot \in \mathrm{PL}$
3. If $A \in \mathrm{PL}$ then $\neg A \in \mathrm{PL}$
4. If $A, B \in \mathrm{PL}$ then $A \wedge B \in \mathrm{PL}$
5. If $A, B \in \mathrm{PL}$ then $A \vee B \in \mathrm{PL}$
6. If $A, B \in \mathrm{PL}$ then $A \rightarrow B \in \mathrm{PL}$
7. If $A \in \mathrm{PL}$ and $x \in V$ then $(\forall x)A \in \mathrm{PL}$
8. If $A \in \mathrm{PL}$ and $x \in V$ then $(\exists x)A \in \mathrm{PL}$

The *logical complexity* of a PL-formula A is the number of logical symbols occurring in A. A formula that has been composed by 1. or 2. only is called *atom formula*. In 7. and 8. the formula A, all of it's subformulas and all terms it contains are said to be *in the scope of* the newly introduced quantifier $(\forall x)$ or $(\exists x)$ respectively. A variable x is called *bound* if it is in the scope of a quantifier of the form (Qx) for $Q \in \{\forall, \exists\}$. A variable is called *free* if it is not bound.

An occurrence λ of a subformula A of a formula F is a string $\in \{1, 2\}^*$ denoting the path to the occurrence of A in (the syntax tree of) F. If λ is an occurrence of A in F we denote this with $F[A]_\lambda$. We say that λ is *empty* iff λ denotes the occurrence of a formula F in itself (λ is the empty path). We will use the notation $|\lambda|$ to denote the (sub-)formula corresponding to the subformula occurrence λ, so for $F[A]_\lambda$ we have $|\lambda| = A$.

Example 2.1. $F = (\forall x)(((\exists y)P(x, y) \wedge Q(x)) \rightarrow Q(x))$ is a PL-formula with a logical complexity of 4. $Q(x)$ occurs twice in F, namely at $\lambda_1 = 112$ and at $\lambda_2 = 12$ ($|\lambda_1| = |\lambda_2| = Q(x)$).

The notation $A[t]$ is used for a formula A that contains the term t. A substitution is a pair consisting of a variable and a term and is written as $\{v \leftarrow t\}$ for a variable v and a term t. By $A\{v \leftarrow t\}$ we mean the formula A where all occurrences of the variable v are replaced by the term t.

2.2 Sequent Calculus

The sequent calculus has been introduced by Gentzen in [7]. The basic structure it operates on is that of a sequent. A sequent essentially consists of two sets of formulas and denotes the claim that the formulas from one set prove one of the formulas from the other set, formally:

Definition 2.4 (Sequent). A sequent is a structure of the form $\Gamma \vdash \Delta$ where Γ and Δ are multisets of PL-formulas. The symbol $\vdash$ serves as a separator.

The interpretation of a sequent $A_1, \ldots, A_n \vdash B_1, \ldots, B_m$ is the formula

$$(A_1 \wedge \ldots \wedge A_n) \rightarrow (B_1 \vee \ldots \vee B_m)$$

A sequent S is derivable iff the formula corresponding to S is valid. For a sequent $\Gamma \vdash \Delta$ we will refer to Γ as the *antecedens* (or *left side*) and to Δ as the *consequent* (or *right side*). If the antecedens is empty it is $\top$ (having the truth value **t**), if the consequent is empty it is $\bot$ (having the truth value **f**). In particular, the derivability of the sequent $\vdash A$ is equivalent to the validity of the formula A and the derivability of the sequent $\Gamma \vdash$ is equivalent to the inconsistency of the (multi-)set of formulas Γ (assuming a sound and complete calculus).

The sequent calculus has two variants: **LK** for classical logic and **LJ** for intuitionistic logic. While definition 2.4 in its general form describes sequents for **LK**, this definition has to be restricted for the calculus **LJ** as follows:

Definition 2.5 (LJ-sequent). An **LJ**-sequent is a sequent of the form $\Gamma \vdash \Delta$ where Γ and Δ are multisets of PL-formulas and $|\Delta| \leq 1$, i.e. Δ either contains one formula or is empty.

The rules for **LJ** differ from the rules of **LK** only in so far that they reflect this restriction on the form of **LJ**-sequents to contain at most one formula in the consequent.

Definition 2.6 (Rule). A rule is an expression of the form

$$\frac{S_1}{S} \qquad \text{or} \qquad \frac{S_1 \quad S_2}{S}$$

where S, S_1, S_2 are sequents. S is called the conclusion of the rule, S_1 and S_2 are called premises.

The intuitive meaning of a rule is that S can be derived if S_1 (and S_2) is (are) given. The first type of rule is called *unary*, the second is called *binary*. In the sequent calculus a sequent is proven by decomposing its formulas from outside in by repeated applications of different rules. We limit ourselves to **LJ** here because we only deal with intuitionistic logic. For a detailed discussion of **LK** see, e.g. [14] or [8].

Definition 2.7 (LJ). **LJ** consists of the following axioms and rules (A and B are formulas, Γ, Π and Δ are multisets of formulas and $|\Delta| \leq 1$):

1. Axioms

$$A \vdash A \quad \text{for a formula A}$$

2. Logical Rules

(a) Disjunction

$$\frac{A, \Gamma \vdash \Delta \quad B, \Pi \vdash \Delta}{A \vee B, \Gamma, \Pi \vdash \Delta} \vee : l \qquad \frac{\Gamma \vdash A}{\Gamma \vdash A \vee B} \vee : r1 \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B} \vee : r2$$

$A \vee B$ is called *main formula*, A and B are called *auxiliary formulas* of the rule $\vee : l$. These definitions are analogous for $\vee : r1$, $\vee : r2$ and the other rules. For an application of the rule $\vee : r1$, the formula B is called *disappearing disjunct* (of the main formula $A \vee B$), for an application of $\vee : r2$, A is called disappearing disjunct.

(b) Conjunction

$$\frac{\Gamma \vdash A \quad \Pi \vdash B}{\Gamma, \Pi \vdash A \wedge B} \wedge : r \qquad \frac{A, \Gamma \vdash \Delta}{A \wedge B, \Gamma \vdash \Delta} \wedge : l1 \qquad \frac{B, \Gamma \vdash \Delta}{A \wedge B, \Gamma \vdash \Delta} \wedge : l2$$

For an application of the rule $\wedge : l1$, the formula B is called *disappearing conjunct* (of the main formula $A \wedge B$), for an application of $\wedge : l2$, A is called disappearing conjunct.

(c) Implication

$$\frac{\Gamma \vdash A \quad B, \Pi \vdash \Delta}{A \rightarrow B, \Gamma, \Pi \vdash \Delta} \rightarrow : l \qquad \frac{A, \Gamma \vdash B}{\Gamma \vdash A \rightarrow B} \rightarrow : r$$

(d) Negation

$$\frac{\Gamma \vdash A}{\neg A, \Gamma \vdash} \neg : l \qquad \frac{A, \Gamma \vdash}{\Gamma \vdash \neg A} \neg : r$$

(e) Universal Quantification

$$\frac{A\{x \leftarrow t\}, \Gamma \vdash \Delta}{(\forall x)A, \Gamma \vdash \Delta} \forall : l \qquad \frac{\Gamma \vdash A\{x \leftarrow \alpha\}}{\Gamma \vdash (\forall x)A} \forall : r$$

For the variable α and the term t the following must hold:

i. t must not contain a variable that occurs bound in A

ii. α is called *eigenvariable* and must not occur in $\Gamma \cup \{A\}$ (*eigenvariable condition*).

The term t (α) is said to be *introduced* by the rule $\forall : l$ ($\forall : r$).

(f) Existential Quantification

$$\frac{\Gamma \vdash A\{x \leftarrow t\}}{\Gamma \vdash (\exists x)A} \exists : r \qquad \frac{A\{x \leftarrow \alpha\}, \Gamma \vdash \Delta}{(\exists x)A, \Gamma \vdash \Delta} \exists : l$$

For the variable α and the term t the following must hold:

i. t must not contain a variable that occurs bound in A

ii. α is called eigenvariable and must not occur in $\Gamma \cup \Delta \cup \{A\}$ (eigenvariable condition).

The term t (α) is said to be *introduced* by the rule $\forall : l$ ($\forall : r$).

3. Structural Rules

(a) Weakening

$$\frac{\Gamma \vdash}{\Gamma \vdash A}\ w : r \qquad \frac{\Gamma \vdash \Delta}{A, \Gamma \vdash \Delta}\ w : l$$

(b) Contraction

$$\frac{A, A, \Gamma \vdash \Delta}{A, \Gamma \vdash \Delta}\ c : l$$

There is no contraction rule for the right side, because **LJ**-sequents are restricted to having at most one formula on the right side, a $c : r$ rule thus could never be applied.

(c) Cut

$$\frac{\Gamma \vdash A \quad A, \Pi \vdash \Delta}{\Gamma, \Pi \vdash \Delta}\ \text{cut}$$

Note that the cut-rule is the only rule whose premises contain a formula that does not occur in the conclusion. Gentzen has shown in his famous Hauptsatz [7] that the cut-rule is redundant, i.e. for every proof using the cut-rule there is a proof that does not use the cut-rule.

Definition 2.8 (LJ-Proof). An **LJ**-proof χ of a sequent S is a tree where the nodes are **LJ**-sequents and the edges are **LJ**-rules connecting their sequents. The root of χ is S and the leaves are axioms.

The *length* of an **LJ**-proof χ (in symbols $l(\chi)$) is defined as the number of edges (rule applications) in χ. The *size* of an **LJ**-proof χ (in symbols $size(\chi)$) is defined as the largest logical complexity of a formula in χ. A *formula occurrence* in a proof χ is a string uniquely denoting the position of a formula in χ. A *subformula occurrence* in a proof is a pair of a formula occurrence (in a proof) and a subformula occurrence (in a formula). For a formula or subformula occurrence θ we write $|\theta|$ to denote the (sub-)formula corresponding to θ.

Example 2.2. An **LJ**-proof:

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{P(a) \vdash P(a) \qquad \dfrac{P(f(a)) \vdash P(f(a))}{P(f(a)) \vdash (\exists y)P(f(y))}\;\exists : r}{P(a), P(a) \rightarrow P(f(a)) \vdash (\exists y)P(f(y))}\;\rightarrow: l}{P(a), (\forall x)(P(x) \rightarrow P(f(x))) \vdash (\exists y)P(f(y))}\;\forall : l}{P(a), P(a) \wedge (\forall x)(P(x) \rightarrow P(f(x))) \vdash (\exists y)P(f(y))}\;\wedge : l2}{P(a) \wedge (\forall x)(P(x) \rightarrow P(f(x))), P(a) \wedge (\forall x)(P(x) \rightarrow P(f(x))) \vdash (\exists y)P(f(y))}\;\wedge : l1}{(P(a) \wedge (\forall x)(P(x) \rightarrow P(f(x)))) \vdash (\exists y)P(f(y))}\;c : l}{\vdash (P(a) \wedge (\forall x)(P(x) \rightarrow P(f(x)))) \rightarrow (\exists y)P(f(y))}\;\rightarrow: r
$$

Example 2.3. The formula $A \vee \neg A$ which is valid in classical logic but not in intuitionistic logic is not derivable in **LJ**. Proving $A \vee \neg A$ in **LJ** would mean proving the sequent $S : \vdash A \vee \neg A$. The only logical rules that can be applied to S obviously are $\vee : r1$ and $\vee : r2$ resulting in the sequents $S_1 : \vdash A$ and $S_2 : \vdash \neg A$ respectively. But in general neither S_1 nor S_2 can be proved. The only structural rule that can be applied is weakening which does not lead to a provable sequent either. Indeed contraction on the right side is necessary to prove $A \vee \neg A$ as shown in the following **LK**-proof:

$$
\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{\vdash A, \neg A}\;\neg : r}{\vdash A, A \vee \neg A}\;\vee : r2}{\vdash A \vee \neg A, A \vee \neg A}\;\vee : r1}{\vdash A \vee \neg A}\;c : r
$$

2.3 Natural Deduction

Natural deduction is a formalism that operates on proofs constructing complicated proofs from simpler ones by repeated applications of different rules. Assumptions play a central role in natural deduction. By introducing an assumption, the proof becomes dependent on it and by discharging the assumption the proof looses this dependency again. For example, if we can prove B by assuming A (i.e. introducing the assumption A), then we can prove $A \rightarrow B$ (by discharging A). To indicate the correspondence between a discharged assumption and the rule that discharges it, we assign a label to every rule that can discharge assumptions and to all assumptions it discharges.

For every subproof χ of a **NJ**-proof ω every assumption γ is in one of two states: It is either *active* or *discharged*. The assumption γ is discharged in χ if it is labeled with some label l and there exists a rule in χ that is also labeled with l. Otherwise it is active. With the notation

$$\begin{array}{c} A \\ \vdots\ \omega \\ B \end{array}$$

we mean a proof ω of B that uses one or more occurrences of A as active assumptions. This proof ω also has a multiset Γ of (other) active assumptions and a multiset Γ' of discharged assumptions. If we write $[\![A]\!]$ instead of A in the proof above we mean that one or more occurrences of A are discharged assumptions in ω.

A natural deduction-rule constructs a proof from one or two simpler proofs. It can therefore be regarded as a *proof constructor*. From this point of view a proof is actually a *proof term* (this will be detailed in section 3.3).

The fundamental symmetry of natural deduction is that of *introduction*- and *elimination*-rules. The first type of rules introduces a new logical symbol while the second type of rule eliminates a logical symbol.

Definition 2.9 (NJ). NJ consists of the following rules (A, B and C are formulas):

1. Conjunction

$$\frac{\begin{array}{c}\vdots \\ A\end{array} \quad \begin{array}{c}\vdots \\ B\end{array}}{A \wedge B} \wedge I \qquad \frac{\begin{array}{c}\vdots \\ A \wedge B\end{array}}{A} \wedge E1 \qquad \frac{\begin{array}{c}\vdots \\ A \wedge B\end{array}}{B} \wedge E2$$

   By using $\wedge I$ we construct a proof of $A \wedge B$ from a proof of A and a proof of B, by using $\wedge E1$ we construct a proof of A from a proof of $A \wedge B$, etc.

2. Disjunction

$$\frac{\begin{array}{c}\vdots \\ A\end{array}}{A \vee B} \vee I1 \qquad \frac{\begin{array}{c}\vdots \\ B\end{array}}{A \vee B} \vee I2 \qquad \frac{\begin{array}{c}\vdots \\ A \vee B\end{array} \quad \begin{array}{c}[\![A]\!]^l \\ \vdots \\ C\end{array} \quad \begin{array}{c}[\![B]\!]^l \\ \vdots \\ C\end{array}}{C} \vee E^l$$

3. Implication

$$\frac{\begin{array}{c}[\![A]\!]^l \\ \vdots \\ B\end{array}}{A \to B} \to I^l \qquad \frac{\begin{array}{c}\vdots \\ A\end{array} \quad \begin{array}{c}\vdots \\ A \to B\end{array}}{B} \to E$$

4. Falsum

$$\frac{\begin{array}{c}\vdots \\ \bot\end{array}}{A} \bot E$$

5. Negation

$$\frac{\begin{matrix}[\![A]\!]^{l}\\ \vdots\\ \bot\end{matrix}}{\neg A}\,\neg I^{l} \qquad \frac{\begin{matrix}\vdots & \vdots\\ A & \neg A\end{matrix}}{\bot}\,\neg E$$

The $\neg I$-rule corresponds to the equivalence of $\neg A$ and $A \rightarrow \bot$ and is - bearing that relationship in mind - a special case of the $\rightarrow I$-rule.

6. Universal Quantification

$$\frac{\begin{matrix}\vdots\\ A\{x \leftarrow \alpha\}\end{matrix}}{(\forall x)A}\,\forall I \qquad \frac{\begin{matrix}\vdots\\ (\forall x)A\end{matrix}}{A\{x \leftarrow t\}}\,\forall E$$

For the variable α and the term t the following must hold:

(a) t must not contain a variable that occurs bound in A

(b) α must not occur in A nor in any active assumption (eigenvariable condition)

7. Existential Quantification

$$\frac{\begin{matrix}\vdots\\ A\{x \leftarrow t\}\end{matrix}}{(\exists x)A}\,\exists I \qquad \frac{\begin{matrix} & [\![A\{x \leftarrow \alpha\}]\!]^{l}\\ \vdots & \vdots\\ (\exists x)A & B\end{matrix}}{B}\,\exists E^{l}$$

For the variable α and the term t the following must hold:

(a) t must not contain a variable that occurs bound in A

(b) α must not occur in A, nor in B nor in any active assumption of the proof of B except those discharged by this $\exists E$-rule.

Example 2.4. An **NJ**-proof:

$$\frac{\dfrac{[\![(\exists x)(\forall y)P(x,y)]\!]^{1} \qquad \dfrac{\dfrac{\dfrac{[\![(\forall y)P(\alpha,y)]\!]^{2}}{P(\alpha,\beta)}\,\forall E}{(\exists x)P(x,\beta)}\,\exists I}{(\forall y)(\exists x)P(x,y)}\,\forall I}{(\forall y)(\exists x)P(x,y)}\,\exists E^{2}}{(\exists x)(\forall y)P(x,y) \rightarrow (\forall y)(\exists x)P(x,y)}\,\rightarrow I^{1}$$

2.4 The Lambda Calculus

The Lambda Calculus is a theory that describes functions as rules (i.e. the intensional aspect of functions). It was originally developed with the aim of providing a general theory of functions and a foundation of mathematics but as it turned out the second goal could not be reached. The main idea of the lambda calculus is to use an abstraction operator to create a function from a term.

Example 2.5. Consider the arithmetical expression $(x + 3)^2$. This expression is just a term but it can also be regarded as a function of x that - given a certain x - calculates the value $(x + 3)^2$. In the lambda calculus this is denoted as $\lambda x.(x + 3)^2$.

2.4.1 Type-Free Lambda Calculus

The type-free lambda calculus is an equality theory over lambda terms. We will only explain the notation briefly, for a detailed treatment of the type-free lambda calculus the interested reader is referred to [5].

Definition 2.10 (Language). The language of the type-free lambda calculus consists of the following elements:

1. A set of *variables*: $\mathsf{V} = \{\mathsf{v}, \mathsf{w}, \mathsf{x}, \mathsf{y}, \mathsf{z}, \mathsf{v}_0, \mathsf{v}_1, \ldots\}$.
2. The *abstractor*: λ
3. Parentheses: (and)

Note the difference in notation between the variables of the lambda calculus $\mathsf{x}, \mathsf{y}, \mathsf{z}, \ldots$ and the variable of first-order logic $x, y, z, \ldots$.

Definition 2.11 (Terms). The set of lambda terms Λ is defined inductively:

1. $\mathsf{x} \in \mathsf{V} \Rightarrow \mathsf{x} \in \Lambda$
2. $\mathsf{x} \in \mathsf{V}, \mathsf{M} \in \Lambda \Rightarrow (\lambda \mathsf{x} \mathsf{M}) \in \Lambda$
3. $M, N \in \Lambda \Rightarrow (MN) \in \Lambda$

We will use the symbol $\equiv$ to denote syntactic equality (as opposed to $=$ denoting equality in the theory λ to be defined below). For a vector of variables $\tilde{\mathsf{x}} \equiv \mathsf{x}_1, \ldots, \mathsf{x}_\mathsf{n}$ we use the notation $\lambda \mathsf{x}_1 \cdots \mathsf{x}_\mathsf{n}.\mathsf{M}$ or $\lambda \tilde{\mathsf{x}}.\mathsf{M}$ for the lambda term $\lambda \mathsf{x}_1(\lambda \mathsf{x}_2(\cdots(\lambda \mathsf{x}_\mathsf{n} \mathsf{M})\ldots))$. For a vector of lambda terms $\tilde{\mathsf{N}} \equiv \mathsf{N}_1, \ldots, \mathsf{N}_\mathsf{n}$ we use the notation $\mathsf{M}\mathsf{N}_1 \ldots \mathsf{N}_\mathsf{n}$ or $\mathsf{M}\tilde{\mathsf{N}}$ for the lambda term $(\cdots((\mathsf{M}\mathsf{N}_1)\mathsf{N}_2)\cdots\mathsf{N}_\mathsf{n})$.

Definition 2.12 (Equality Theory). The equality theory λ is axiomatized by the following axiom and rule schemata. Let $\mathsf{M}, \mathsf{N}, \mathsf{L} \in \Lambda, \mathsf{x} \in \mathsf{V}$:

1. Reflexivity: $\mathsf{M} = \mathsf{M}$
2. Symmetry: $\mathsf{M} = \mathsf{N} \Rightarrow \mathsf{N} = \mathsf{M}$
3. Transitivity: $\mathsf{M} = \mathsf{N}, \mathsf{N} = \mathsf{L} \Rightarrow \mathsf{M} = \mathsf{L}$
4. β-Conversion: $(\lambda \mathsf{x}.\mathsf{M})\mathsf{N} = \mathsf{M}\{\mathsf{x} \leftarrow \mathsf{N}\}$
   $\mathsf{M}\{\mathsf{x} \leftarrow \mathsf{N}\}$ denotes the substitution of N for all occurrences of x in M
5. Substitutivity 1: $\mathsf{M} = \mathsf{N} \Rightarrow \mathsf{ML} = \mathsf{NL}$
6. Substitutivity 2: $\mathsf{M} = \mathsf{N} \Rightarrow \mathsf{LM} = \mathsf{LN}$
7. (rule ξ): $\mathsf{M} = \mathsf{N} \Rightarrow \lambda \mathsf{x}.\mathsf{M} = \lambda \mathsf{x}.\mathsf{N}$

2.4.2 Typed Lambda Calculus

In introducing typed lambda calculus we will essentially follow the approach in [6] (with a slightly different notation) and extend it by the inclusion of first-order conjunction ($\wedge$) and universal quantification ($\forall$) like it can be found in [9] and [12].

Definition 2.13 (PL^-). The set $\mathrm{PL}^- \subset \mathrm{PL}$ is the set of all PL-formulas containing $\wedge, \rightarrow, \forall$ and $\perp$ as only logical symbols.

Definition 2.14 (Types). The set of types $\mathbb{T}$:

1. $i \in \mathbb{T}$
2. $A \in \mathrm{PL}^- \Rightarrow A \in \mathbb{T}$

The type i is intended to denote the type of individuals, i.e. the set of terms of first-order logic.

Definition 2.15 (Language). The language of the typed lambda calculus consists of the following elements:

1. For each type $T \in \mathbb{T}$ a set of variables $\mathsf{V}_T = \{\mathsf{x}^T, \mathsf{y}^T, \mathsf{z}^T \ldots\}$
2. The abstractor: λ
3. Pair brackets: $\langle$ and $\rangle$
4. Projections: π^1 and π^2
5. Parentheses: $($, $)$

The set of all variables is $\mathsf{V} = \bigcup_{T \in \mathbb{T}} V_T$. We will use the notation $\mathsf{M} : A$ for a lambda term M and a type A to indicate that M has the type A. To simplify the notation we will sometimes omit the type superscript for variables, so for example $\mathsf{x}^A : A$ may be written as $\mathsf{x} : A$. When writing down large terms we will specify the types of the variables separated from the term to increase readability.

Definition 2.16 ($\Lambda_{\mathbb{T}}$). The set of typed lambda terms $\Lambda_{\mathbb{T}}$ is defined inductively by the following axioms and rules (let $x \in V$, $t \in T$, $F \in \mathrm{PL}^-$, $\mathsf{x} \in \mathsf{V}$, $A, B \in \mathbb{T}$, $\mathsf{M}, \mathsf{N} \in \Lambda_{\mathbb{T}}$):

1. Axiom

$$\mathsf{x}^A : A$$

2. Formula-Abstraction ($\rightarrow$-Introduction)

$$\frac{\mathsf{x} : F \quad \mathsf{M} : A}{\lambda \mathsf{x}.\mathsf{M} : F \rightarrow A}$$

3. Formula-Application ($\rightarrow$-Elimination)

$$\frac{\mathsf{N} : F \quad \mathsf{M} : F \rightarrow A}{\mathsf{MN} : A}$$

4. i-Abstraction ($\forall$-Introduction)

$$\frac{\mathsf{x} : i \quad \mathsf{M}\{\mathsf{x} \leftarrow \alpha\} : A\{x \leftarrow \alpha\}}{\lambda \mathsf{x}.\mathsf{M} : (\forall x)A}$$

if α does not occur in A nor in the type of a free variable of M.

5. i-Application ($\forall$-Elimination)

$$\frac{t : i \quad \mathsf{M} : (\forall x)A}{\mathsf{M}t : A\{x \leftarrow t\}}$$

if t does not contain a variable that is bound in A

6. Pair Construction ($\wedge$-Introduction)

$$\frac{\mathsf{M} : A \quad \mathsf{N} : B}{\langle \mathsf{M}, \mathsf{N} \rangle : A \wedge B}$$

7. Projection ($\wedge$-Elimination)

$$\frac{\mathsf{M} : A \wedge B}{\pi^1 \mathsf{M} : A} \qquad \frac{\mathsf{M} : A \wedge B}{\pi^2 \mathsf{M} : B}$$

The names of these rules already suggest the relation between **NJ** and $\lambda_{\mathbb{T}}$ that will be detailed in section 3.3.

For notational convenience we will sometimes use the shortcut $\Lambda_A = \{\mathsf{M} \in \Lambda_{\mathbb{T}} \mid \mathsf{M} : A\}$. Also note that there are two different notations for the same i-variable: x (as part of a PL-formula) and x^i (as part of a $\lambda_{\mathbb{T}}$-term). We will identify x and x^i to facilitate the mapping between the i-variable in the type and the i-variable in the lambda term of the form $\lambda\mathsf{x}.\mathsf{M} : (\forall x)A$.

Example 2.6. This example illustrates the construction of a typed lambda term

$$\dfrac{f(a) : i \qquad \dfrac{\mathsf{x} : i \qquad \dfrac{\mathsf{y} : i \qquad \dfrac{\mathsf{M} : P(\alpha,\beta) \qquad \mathsf{N} : Q(\alpha)}{\langle \mathsf{M}, \mathsf{N}\rangle : P(\alpha,\beta) \wedge Q(\alpha)}}{\lambda\mathsf{y}.\langle \mathsf{M}, \mathsf{N}\rangle : (\forall y)(P(\alpha, y) \wedge Q(\alpha))}}{\lambda\mathsf{xy}.\langle \mathsf{M}, \mathsf{N}\rangle : (\forall x)(\forall y)(P(x, y) \wedge Q(x))}}{(\lambda\mathsf{xy}.\langle \mathsf{M}, \mathsf{N}\rangle)f(a) : (\forall y)(P(f(a), y) \wedge Q(f(a)))}$$

Chapter 3

Translations

Both, **LJ** and **NJ** describe intuitionistic logic, so the question wether and - if yes - how these formalisms can be translated into each other is natural to ask. In this chapter we will answer this question positively in both directions and specify the corresponding translations. In section 3.3 we will introduce the Curry-Howard Isomorphism which describes a close correspondence between the typed lambda calculus and natural deduction and provides an operational interpretation of (a subset of) intuitionistic logic.

In the following we will use the notion of calculus simulation: A calculus $\mathfrak{A}$ *simulates* a calculus $\mathfrak{B}$ if there exists a computable transformation T from derivations in $\mathfrak{B}$ to derivations in $\mathfrak{A}$ such that: If π is a proof of a formula F in $\mathfrak{B}$ then $T(\pi)$ is a proof of F in $\mathfrak{A}$. For a class of functions K we say that a calculus $\mathfrak{A}$ *K-simulates* a calculus $\mathfrak{B}$ if $\mathfrak{A}$ simulates $\mathfrak{B}$ and the proof transformation T is computable in time t for a $t \in K$. An important class of functions in this context is the class of all polynomials P, so P-simulation means simulation in polynomial time.

In this chapter we will show that **LJ** and **NJ** P-simulate each other by defining appropriate proof transformations. These proof transformations have another property that is not directly related to complexity: They are "quasi homomorphic" in the sense that a rule in one calculus is directly translated into a sequence of rules in the other calculus without modifying the translations of the proof(s) above it.

3.1 Simulating LJ in NJ

In this section we will prove that **NJ** simulates **LJ** by defining a translation of **LJ**-proofs to **NJ**-proofs. This translation can also be found in [15].

Theorem 3.1 (NJ simulates LJ). For every **LJ**-proof χ of a sequent $\Gamma \vdash \Delta$ there exists an **NJ**-proof φ of F with $\Delta = \{F\}$ or (if $\Delta = \emptyset$) $F = \bot$

and Γ being exactly the active assumptions of φ.

Proof. We define a mapping Ψ translating an **LJ**-proof χ into an **NJ**-proof $\Psi(\chi)$. The following table defines Ψ by induction on the structure of χ. An entry of the form $\chi \Longrightarrow \varphi$ should be read as: $\Psi(\chi) := \varphi$.

$$A \vdash A \quad \Longrightarrow \quad A$$

$$\dfrac{\begin{matrix}\vdots\ \chi_1 \\ \Gamma \vdash A\end{matrix} \qquad \begin{matrix}\vdots\ \chi_2 \\ \Pi \vdash B\end{matrix}}{\Gamma, \Pi \vdash A \wedge B}\ \wedge : r \quad \Longrightarrow \quad \dfrac{\begin{matrix}\Gamma \\ \vdots\ \Psi(\chi_1) \\ A\end{matrix} \qquad \begin{matrix}\Pi \\ \vdots\ \Psi(\chi_2) \\ B\end{matrix}}{A \wedge B}\ \wedge I$$

$$\dfrac{\begin{matrix}\vdots\ \chi_1 \\ A, \Gamma \vdash \Delta\end{matrix}}{A \wedge B, \Gamma \vdash \Delta}\ \wedge : l1 \quad \Longrightarrow \quad \begin{matrix}\dfrac{A \wedge B}{A}\ \wedge E1 \quad \Gamma \\ \vdots\ \Psi(\chi_1) \\ F\end{matrix}$$

for $\Delta = \{F\}$ or (if $\Delta = \emptyset$) $F = \bot$

$$\dfrac{\begin{matrix}\vdots\ \chi_1 \\ B, \Gamma \vdash \Delta\end{matrix}}{A \wedge B, \Gamma \vdash \Delta}\ \wedge : l2 \quad \Longrightarrow \quad \begin{matrix}\dfrac{A \wedge B}{B}\ \wedge E2 \quad \Gamma \\ \vdots\ \Psi(\chi_1) \\ F\end{matrix}$$

for $\Delta = \{F\}$ or (if $\Delta = \emptyset$) $F = \bot$

$$\dfrac{\begin{matrix}\vdots\ \chi_1 \\ A, \Gamma \vdash C\end{matrix} \qquad \begin{matrix}\vdots\ \chi_2 \\ B, \Pi \vdash C\end{matrix}}{A \vee B, \Gamma, \Pi \vdash C}\ \vee : l \quad \Longrightarrow \quad \dfrac{A \vee B \qquad \begin{matrix}[\![A]\!]^l \quad \Gamma \\ \vdots\ \Psi(\chi_1) \\ C\end{matrix} \qquad \begin{matrix}[\![B]\!]^l \quad \Pi \\ \vdots\ \Psi(\chi_2) \\ C\end{matrix}}{C}\ \vee E^l$$

where l is a label not occurring in $\Psi(\chi_1)$ nor in $\Psi(\chi_2)$

$$\dfrac{\begin{matrix}\vdots\ \chi_1 \\ \Gamma \vdash A\end{matrix}}{\Gamma \vdash A \vee B}\ \vee : r1 \quad \Longrightarrow \quad \dfrac{\begin{matrix}\Gamma \\ \vdots\ \Psi(\chi_1) \\ A\end{matrix}}{A \vee B}\ \vee I1$$

$$\dfrac{\begin{matrix}\vdots\ \chi_1 \\ \Gamma \vdash B\end{matrix}}{\Gamma \vdash A \vee B}\ \vee : r2 \quad \Longrightarrow \quad \dfrac{\begin{matrix}\Gamma \\ \vdots\ \Psi(\chi_1) \\ B\end{matrix}}{A \vee B}\ \vee I2$$

$$\dfrac{\begin{matrix}\vdots\ \chi_1 \\ A, \Gamma \vdash B\end{matrix}}{\Gamma \vdash A \to B}\ \to: r \quad \Longrightarrow \quad \dfrac{\begin{matrix}[\![A]\!]^l \quad \Gamma \\ \vdots\ \Psi(\chi_1) \\ B\end{matrix}}{A \to B}\ \to I^l$$

where l is a label not occurring in $\Psi(\chi_1)$

$$\dfrac{\overset{\vdots\,\chi_1}{\Gamma \vdash A} \qquad \overset{\vdots\,\chi_2}{B, \Pi \vdash \Lambda}}{A \to B, \Gamma, \Pi \vdash \Lambda} \to: l \qquad \Longrightarrow \qquad \begin{array}{c} \dfrac{\overset{\Gamma}{\overset{\vdots\,\Psi(\chi_1)}{A}} \qquad A \to B}{B} \to E \quad \Pi \\ \vdots\,\Psi(\chi_2) \\ \Lambda \end{array}$$

$$\dfrac{\overset{\vdots\,\chi_1}{A, \Gamma \vdash}}{\Gamma \vdash \neg A} \neg : r \qquad \Longrightarrow \qquad \dfrac{\overset{[\![A]\!]^l \quad \Gamma}{\overset{\vdots\,\Psi(\chi_1)}{\bot}}}{\neg A} \neg I^l$$

where l is a label not occurring in $\Psi(\chi_1)$.

$$\dfrac{\overset{\vdots\,\chi_1}{\Gamma \vdash A}}{\Gamma, \neg A \vdash} \neg : l \qquad \Longrightarrow \qquad \dfrac{\overset{\Gamma}{\overset{\vdots\,\Psi(\chi_1)}{A}} \qquad \neg A}{\bot} \neg E$$

$$\dfrac{\overset{\vdots\,\chi_1}{\Gamma \vdash A\{x \leftarrow \alpha\}}}{\Gamma \vdash (\forall x)A} \forall : r \qquad \Longrightarrow \qquad \dfrac{\overset{\Gamma}{\overset{\vdots\,\Psi(\chi_1)}{A\{x \leftarrow \alpha\}}}}{(\forall x)A} \forall I$$

This translation preserves the eigenvariable condition.

$$\dfrac{\overset{\vdots\,\chi_1}{A\{x \leftarrow t\}, \Gamma \vdash \Delta}}{(\forall x)A, \Gamma \vdash \Delta} \forall : l \qquad \Longrightarrow \qquad \begin{array}{c} \dfrac{(\forall x)A}{A\{x \leftarrow t\}} \forall E \quad \Gamma \\ \vdots\,\Psi(\chi_1) \\ F \end{array}$$

for $\Delta = \{F\}$ or (if $\Delta = \emptyset$) $F = \bot$

$$\dfrac{\overset{\vdots\,\chi_1}{\Gamma \vdash A\{x \leftarrow t\}}}{\Gamma \vdash (\exists x)A} \exists : r \qquad \Longrightarrow \qquad \dfrac{\overset{\Gamma}{\overset{\vdots\,\Psi(\chi_1)}{A\{x \leftarrow t\}}}}{(\exists x)A} \exists I$$

$$\dfrac{\overset{\vdots\,\chi_1}{A\{x \leftarrow \alpha\}, \Gamma \vdash \Delta}}{(\exists x)A, \Gamma \vdash \Delta} \exists : l \qquad \Longrightarrow \qquad \dfrac{(\exists x)A \qquad \overset{[\![A\{x \leftarrow \alpha]\}]\!]^l \quad \Gamma}{\overset{\vdots\,\Psi(\chi_1)}{F}}}{F} \exists E^l$$

for $\Delta = \{F\}$ or (if $\Delta = \emptyset$) $F = \bot$ and where l is a label not occurring in $\Psi(\chi_1)$
This translation preserves the eigenvariable condition.

$$\dfrac{\begin{array}{c}\vdots\ \chi_1\\ \Gamma \vdash\end{array}}{\Gamma \vdash A}\ w:r \qquad\Longrightarrow\qquad \dfrac{\begin{array}{c}\Gamma\\ \vdots\ \Psi(\chi_1)\\ \bot\end{array}}{A}\ \bot$$

$$\dfrac{\begin{array}{c}\vdots\ \chi_1\\ \Gamma \vdash \Delta\end{array}}{A, \Gamma \vdash \Delta}\ w:l \qquad\Longrightarrow\qquad \begin{array}{c}A \quad \Gamma\\ \vdots\ \Psi(\chi_1)\\ F\end{array}$$

for $\Delta = \{F\}$ or (if $\Delta = \emptyset$) $F = \bot$ and where l is a label not occurring in $\Psi(\chi_1)$

$$\dfrac{\begin{array}{c}\vdots\ \chi_1\\ A, A, \Gamma \vdash \Delta\end{array}}{A, \Gamma \vdash \Delta}\ c:l \qquad\Longrightarrow\qquad \begin{array}{c}A \quad \Gamma\\ \vdots\ \Psi(\chi_1)\\ F\end{array}$$

$$\dfrac{\begin{array}{c}\vdots\ \chi_1\\ \Gamma \vdash A\end{array} \qquad \begin{array}{c}\vdots\ \chi_2\\ A, \Pi \vdash \Delta\end{array}}{\Gamma, \Pi \vdash \Delta}\ cut \qquad\Longrightarrow\qquad \begin{array}{c}\begin{array}{c}\Gamma\\ \vdots\ \Psi(\chi_1)\\ A\end{array} \quad \Pi\\ \vdots\ \Psi(\chi_2)\\ F\end{array}$$

for $\Delta = \{F\}$ or (if $\Delta = \emptyset$) $F = \bot$

It can be shown by induction that Ψ satisfies the following property: If χ is an **LJ**-proof of $\Gamma \vdash F$ then $\Psi(\chi)$ is an **NJ**-proof of F from the active assumptions Γ. □

Example 3.1. Consider the following **LJ**-proof

$\chi =$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{P(a) \vdash P(a) \qquad \dfrac{P(f(a)) \vdash P(f(a)) \qquad P(f(f(a))) \vdash P(f(f(a)))}{P(f(a)), P(f(a)) \to P(f(f(a))) \vdash P(f(f(a)))}\ \to: l}{P(a), P(a) \to P(f(a)), P(f(a)) \to P(f(f(a))) \vdash (P(f(f(a)))}\ \to: l}{P(a), P(a) \to P(f(a)), (\forall x)(P(x) \to P(f(x)) \vdash P(f(f(a)))}\ \forall: l}{P(a), (\forall x)(P(x) \to P(f(x)), (\forall x)(P(x) \to P(f(x)) \vdash P(f(f(a)))}\ \forall: l}{P(a), (\forall x)(P(x) \to P(f(x)) \vdash P(f(f(a)))}\ c:l}{P(a) \vdash (\forall x)(P(x) \to P(f(x)) \to P(f(f(a)))}\ \to: r}{\vdash P(a) \to ((\forall x)(P(x) \to P(f(x))) \to P(f(f(a))))}\ \to: r$$

The translation $\Psi(\chi) =$

$$\dfrac{\dfrac{\llbracket P(a)\rrbracket^2 \quad \dfrac{\llbracket(\forall x)(P(x)\to P(f(x))\rrbracket^1}{P(a)\to P(f(a))}\,\forall E}{P(f(a))}\to E \qquad \dfrac{\llbracket(\forall x)(P(x)\to P(f(x))\rrbracket^1}{P(f(a))\to P(f(f(a)))}\,\forall E}{\dfrac{P(f(f(a)))}{\dfrac{(\forall x)(P(x)\to P(f(x))\to P(f(f(a)))}{P(a)\to((\forall x)(P(x)\to P(f(x)))\to P(f(f(a))))}\to I^2}\to I^1}\to E$$

3.2 Simulating NJ in LJ

We will now prove that **LJ** simulates **NJ** by describing an appropriate translation which is essentially the one given by Gentzen in [7].

Theorem 3.2 (LJ simulates NJ). For every **NJ**-proof φ of a formula F from the active assumptions Γ there exists an **LJ**-proof of the sequent $\Gamma \vdash F$.

Proof. Let φ be an **NJ**-proof of a formula F from the (multiset of) active assumptions Γ. We obtain the derivation φ' from φ by replacing every formula A in φ by the sequent $\Gamma \vdash A$ where Γ is the multiset of active assumptions in the proof of A. After that, replace all sequents of the from $\Gamma \vdash \bot$ by $\Gamma \vdash$. The derivation φ' is now a tree of sequences where the leafs are **LJ**-axioms and the root is $\Gamma \vdash F$.[1]

We obtain φ'' by carrying out the following rule renamings: $\vee I \hookrightarrow \vee :\ r, \wedge I \hookrightarrow \wedge : r, \exists I \hookrightarrow \exists : r, \forall I \hookrightarrow \forall : r, \bot \hookrightarrow w : r$.

The following table defines the transformation of the derivation φ'' into an **LJ**-proof:

$$\dfrac{\begin{array}{c}\vdots\ \chi\\ \Gamma, A, \ldots, A \vdash B\end{array}}{\Gamma \vdash A \to B}\to I^l \qquad \Longrightarrow \qquad \dfrac{\dfrac{\begin{array}{c}\vdots\ \chi\\ \Gamma, A, \ldots, A \vdash B\end{array}}{\Gamma, A \vdash B}\, c : l^*}{\Gamma \vdash A \to B}\to : r$$

$$\dfrac{\begin{array}{c}\vdots\ \chi\\ \Gamma, A, \ldots, A \vdash\end{array}}{\Gamma \vdash \neg A}\neg I^l \qquad \Longrightarrow \qquad \dfrac{\dfrac{\begin{array}{c}\vdots\ \chi\\ \Gamma, A, \ldots, A \vdash\end{array}}{\Gamma, A \vdash}\, c : l^*}{\Gamma \vdash \neg A}\neg : r$$

[1] Note that φ' is neither an **NJ**-proof nor an **LJ**-proof but something "in-between".

$$\dfrac{\overset{\vdots\,\chi}{\Gamma \vdash A \wedge B}}{\Gamma \vdash A}\wedge E1 \quad\Longrightarrow\quad \dfrac{\overset{\vdots\,\chi}{\Gamma \vdash A \wedge B} \quad \dfrac{A \vdash A}{A \wedge B \vdash A}\wedge : l1}{\Gamma \vdash A}\,cut$$

$$\dfrac{\overset{\vdots\,\chi}{\Gamma \vdash A \wedge B}}{\Gamma \vdash B}\wedge E2 \quad\Longrightarrow\quad \dfrac{\overset{\vdots\,\chi}{\Gamma \vdash A \wedge B} \quad \dfrac{B \vdash B}{A \wedge B \vdash B}\wedge : l2}{\Gamma \vdash B}\,cut$$

$$\dfrac{\overset{\vdots\,\chi_1}{\Gamma \vdash A} \quad \overset{\vdots\,\chi_2}{\Delta \vdash A \to B}}{\Gamma, \Delta \vdash B}\to E \quad\Longrightarrow\quad \dfrac{\overset{\vdots\,\chi_2}{\Delta \vdash A \to B} \quad \dfrac{\overset{\vdots\,\chi_1}{\Gamma \vdash A} \quad B \vdash B}{\Gamma, A \to B \vdash B}\to l}{\Gamma, \Delta \vdash B}\,cut$$

$$\dfrac{\overset{\vdots\,\chi_1}{\Gamma \vdash A} \quad \overset{\vdots\,\chi_2}{\Delta \vdash \neg A}}{\Gamma, \Delta \vdash}\neg E \quad\Longrightarrow\quad \dfrac{\overset{\vdots\,\chi_2}{\Delta \vdash \neg A} \quad \dfrac{\overset{\vdots\,\chi_1}{\Gamma \vdash A}}{\neg A, \Gamma \vdash}\neg : l}{\Gamma, \Delta \vdash}\,cut$$

$$\dfrac{\overset{\vdots\,\chi}{\Gamma \vdash (\forall x)A}}{\Gamma \vdash A\{x \leftarrow t\}}\forall E \quad\Longrightarrow\quad \dfrac{\overset{\vdots\,\chi}{\Gamma \vdash (\forall x)A} \quad \dfrac{A\{x \leftarrow t\} \vdash A\{x \leftarrow t\}}{(\forall x)A \vdash A\{x \leftarrow t\}}\forall : l}{\Gamma \vdash A\{x \leftarrow t\}}\,cut$$

$$\dfrac{\overset{\vdots\,\psi}{\Gamma \vdash A \vee B} \quad \overset{\vdots\,\chi_1}{\Delta_1, A, \ldots, A \vdash C} \quad \overset{\vdots\,\chi_2}{\Delta_2, B, \ldots, B \vdash C}}{\Gamma, \Delta_1, \Delta_2 \vdash C}\vee E$$

$$\Longrightarrow\quad \dfrac{\overset{\vdots\,\psi}{\Gamma \vdash A \vee B} \quad \dfrac{\dfrac{\overset{\vdots\,\chi_1}{\Delta_1, A, \ldots, A \vdash C}}{\Delta_1, A \vdash C}c : l^* \quad \dfrac{\overset{\vdots\,\chi_2}{\Delta_2, B, \ldots, B \vdash C}}{\Delta_2, B \vdash C}c : l^*}{\Delta_1, \Delta_2, A \vee B \vdash C}\vee : l}{\Gamma, \Delta_1, \Delta_2 \vdash C}\,cut$$

$$\dfrac{\overset{\vdots\,\chi_1}{\Gamma \vdash (\exists x)A} \quad \overset{\vdots\,\chi_2}{\Delta, A\{x \leftarrow \alpha\}, \ldots, A\{x \leftarrow \alpha\} \vdash C}}{C}\exists E^l$$

$$\Longrightarrow\quad \dfrac{\overset{\vdots\,\chi_1}{\Gamma \vdash (\exists x)A} \quad \dfrac{\dfrac{\overset{\vdots\,\chi_2}{\Delta, A\{x \leftarrow \alpha\}, \ldots, A\{x \leftarrow \alpha\} \vdash C}}{\Delta, A\{x \leftarrow \alpha\} \vdash C}c : l^*}{\Delta, (\exists x)A \vdash C}\exists : l}{\Gamma, \Delta \vdash C}\,cut$$

This translation preserves the eigenvariable condition.

By applying these transformations we obtain an **LJ**-proof of the sequent

$\Gamma \vdash F$ where F is the formula proven by the original **NJ**-proof φ and Γ are the active assumptions of φ. $\square$

3.3 The Curry-Howard Isomorphism

The Curry-Howard Isomorphism is a bijection between **NJ**-proofs (of PL^--formulas) and typed lambda terms (as defined in Section 2.4.2). We speak about an isomorphism and not about a bijection because the notions of normalization[2] on both sides correspond perfectly. We use the notation $\varphi \simeq \mathsf{M}$ for an **NJ**-proof φ and a typed lambda term M to indicate that φ corresponds to M via the Curry-Howard isomorphism. The isomorphism is described in the following table:

A	$\simeq$	x^A	
$[\![A]\!]^l$	$\simeq$	x_l^A	
$\dfrac{\overset{\vdots\,\varphi_1}{A} \qquad \overset{\vdots\,\varphi_2}{B}}{A \wedge B}\,\wedge I$	$\simeq$	$\langle \mathsf{M}_1, \mathsf{M}_2 \rangle$	if $\varphi_1 \simeq \mathsf{M}_1$ and $\varphi_2 \simeq \mathsf{M}_2$
$\dfrac{\overset{\vdots\,\varphi}{A \wedge B}}{A}\,\wedge E1$	$\simeq$	$\pi^1(\mathsf{M})$	if $\varphi \simeq \mathsf{M}$
$\dfrac{\overset{\vdots\,\varphi}{A \wedge B}}{B}\,\wedge E2$	$\simeq$	$\pi^2(\mathsf{M})$	if $\varphi \simeq \mathsf{M}$
$\dfrac{\overset{[\![A]\!]^l}{\overset{\vdots\,\varphi}{B}}}{A \to B}\,\to I^l$	$\simeq$	$\lambda \mathsf{x}_l^A.\mathsf{M}$	if $\varphi \simeq \mathsf{M}$
$\dfrac{\overset{\vdots\,\varphi_1}{A} \qquad \overset{\vdots\,\varphi_2}{A \to B}}{B}\,\to E$	$\simeq$	$\mathsf{M}_2\mathsf{M}_1$	if $\varphi_1 \simeq \mathsf{M}_1$ and $\varphi_2 \simeq \mathsf{M}_2$
$\dfrac{\overset{\vdots\,\varphi}{A\{x \leftarrow \alpha\}}}{(\forall x)A}\,\forall I$	$\simeq$	$\lambda \mathsf{x}^i.\mathsf{M}$	if $\varphi \simeq \mathsf{M}$

[2]Normalization will be defined in Section 4.2.

$$\dfrac{\begin{array}{c}\vdots\ \varphi\\ (\forall x)A\end{array}}{A\{x \leftarrow t\}}\ \forall E \qquad \simeq \qquad \mathsf{M}t \qquad \text{if } \varphi \simeq \mathsf{M}$$

3.4 Discussion

The translations presented above shed some light on the relation and the differences between **LJ** and **NJ**. The most important difference is probably the nature of a rule in these two systems. In **LJ** a rule is something like a "meta-implication": The rule

$$\frac{\Gamma \vdash \Delta}{\Gamma' \vdash \Delta'}$$

tells us that if we are given the sequent $\Gamma \vdash \Delta$ we can derive $\Gamma' \vdash \Delta'$. This holds always, regardless of the size or form of the proof of $\Gamma \vdash \Delta$. The rules in natural deduction have a much more constructive meaning. In fact, one cannot regard these rules as connecting two (or three, for the binary case) formulas, rather it is necessary to regard a rule as "proof constructor" and so proofs are actually proof terms. This proofs-as-terms view becomes even more evident due to the Curry-Howard Isomorphism. Maybe typed lambda terms are an even more appropriate description of the underlying objects than **NJ**-proofs because the **NJ**-rules create a false sense of a flat tree structure (like **LJ**) where actually there is a term structure. The difference between a flat tree structure and a term structure is wether some (interesting) meaning can be assigned to a single node. In the flat tree structure of an **LJ**-proof we know that each node is a *valid* sequent. A node in an **NJ**-proof is a formula which we know nothing about. In the term structure of an **NJ**-proof only a subtree has an interesting meaning: It is a proof of the formula at the root from all open assumptions. This flat tree vs. term structure becomes even more evident if we consider the assumptions in **NJ** and the rules which discharge assumptions ($\vee E, \rightarrow I, \neg I, \exists E$). Such a rule can not be described only locally, instead it is necessary to know the whole proof above this rule (the subtree) in order to apply it.

Concerning the correspondence of the proofs it can be said that the active assumptions in an **NJ**-proof correspond to the left side of a sequent and the formula at the root of the **NJ**-proof corresponds to the right side of the sequent. The fundamental symmetry in **NJ** is that of introduction vs. elimination rules whereas the fundamental symmetry in **LJ** is that of left-rules vs. right-rules. This can be seen for example in the translation of **NJ**-proofs to **LJ**-proofs where the introduction rules (almost) directly correspond to the right rules and the elimination rules correspond to the left rules and a cut. Using cuts for the elimination rules in this translation is necessary because the elimination rules decrease the logical complexity of formulas (when viewed upside-down in an **NJ**-proof-tree) and **LJ** has no means to do this except with the cut rule. This property of the elimination rules also has the consequence that there is no subformula property[3] in **NJ**. This makes **NJ** particularly unsuited for automated proof search.

Of course, most of what has been said here also applies to the classical counterparts of these system: **LK** and **NK**.

[3] A proof system has the *subformula property* if every formula that occurs in a proof of F is a subformula of F (modulo term instantiations).

Chapter 4

Cut-Elimination and Normalization

In this chapter we will investigate reductions in the formal systems presented above: *Cut-Elimination* in the sequent calculus and *Normalization* in natural deduction and in the typed lambda calculus.

4.1 Cut-Elimination

The cut-rule is in some ways different than the other rules of the sequent calculus **LJ** (see section 2.2). First it is the only rule in which new formulas occur (when viewed from bottom-up) or when viewed top-down the only rule in which not all formulas have successors. The second difference is that the cut-rule is redundant, in the sense that for every proof we can find another proof of the same end-sequent which does not use the cut-rule. This second result is the cut-elimination theorem or "Hauptsatz" of Gentzen. This theorem has important implications that we will discuss later.

If we know that the cut-rule is redundant, the question arises why we should consider the cut-rule at all. What is the interest in a redundant rule ? The interest in the cut-rule is mainly motivated by the fact that a cut is a natural formalization of the use of lemmas in a proof. In a cut of the form

$$\dfrac{\begin{array}{c}\vdots\ \chi_1 \\ \Gamma \vdash A\end{array} \qquad \begin{array}{c}\vdots\ \chi_2 \\ A, \Pi \vdash \Delta\end{array}}{\Gamma, \Pi \vdash \Delta}\ cut$$

the formula A can be seen as a lemma, χ_1 as its proof (from Γ) and the cut rule as application of the lemma A to derive Δ from Π and Γ. The cut-rule is also very powerful from the point of view of proof complexity. Statman [11] and Orevkov [10] independently have shown that cut-elimination is of non-elementary complexity, i.e. if the original proof (containing cuts) has a length of n then the cut-free proof may grow up to a length of

$$\underbrace{2^{\cdot^{\cdot^{\cdot^{2}}}}}_{\text{n times}}$$

In this section we will sketch Gentzen's proof of the "Hauptsatz". It is a constructive proof so an algorithm for cut-elimination can be extracted. A more general form of this algorithm is to regard it as a deterministic interpretation of a set of rules like it is done in [4]. Another deterministic interpretation of this set of rules is the cut-elimination algorithm of Tait [13]. The interest in (speeding up) cut-elimination algorithms is motivated by the possibility of automated proof analysis and led to algorithms such as Cut-Projection [1],[2] that is the topic of this thesis and CERES (cut elimination and redundancy elimination by resolution, see [3]).

Definition 4.1 (Degree). The *degree* of a cut is the logical complexity of the cut formula.

Definition 4.2 (Rank). Let χ be a proof segment of the form

$$\dfrac{\begin{array}{c}\vdots\ \chi_1\\ \Gamma \vdash A\end{array} \qquad \begin{array}{c}\vdots\ \chi_2\\ A, \Pi \vdash \Delta\end{array}}{\Gamma, \Pi \vdash \Delta}\ cut$$

The *left-rank* of this cut is the number of sequents in χ_1 that contain A as a predecessor of the occurrence of A in the end-sequent of χ_1. The *right-rank* is defined analogously. The *rank* of this cut is the sum of left-rank and right-rank.

Definition 4.3 (Mix). A *mix* is the following **LJ**-rule:

$$\dfrac{\Gamma \vdash A \quad \Pi \vdash \Delta}{\Gamma, \Pi^* \vdash \Delta}\ mix(A)$$

where $A \in \Pi$ and $\Pi^* = \Pi \setminus \{A\}$.

To simplify the proof Gentzen uses mix-rules instead of cut-rules. The difference between a cut and a mix is that in a mix *all* occurrences of A are deleted from Π whereas in a cut only *one* occurrence is deleted. Obviously every cut can be transformed into a mix with succeeding weakening rules and vice versa: Every mix can be transformed into a cut with preceeding contraction rules. The definitions of rank and degree can of course also be used for mixes.

Theorem 4.1 (Hauptsatz). For every **LJ**-proof χ, it is possible to construct an **LJ**-proof χ' of the same end-sequent that does not contain mixes.

Proof Sketch. Gentzen's proof proceeds by considering an uppermost mix ξ and eliminating it by creating a mix-free proof that has the same endsequent as the conclusion sequent of ξ. A mix is eliminated by replacing it with one or more mixes with lower rank. If the uppermost mix has (the smallest possible) rank 2, the degree is reduced by one. This procedure is iterated until the proof is free of mixes. It suffices to focus on the elimination of a mix in a proof ψ of the form

$$\dfrac{\begin{array}{c}\vdots\,\psi_1\\ \Gamma \vdash A\end{array} \quad \begin{array}{c}\vdots\,\psi_2\\ \Pi \vdash \Delta\end{array}}{\Gamma, \Pi^* \vdash \Delta}\ mix(A)$$

where ψ_1 and ψ_2 do not contain mixes. In the following we sketch the cut-elimination procedure for **LJ** by listing some of the rules from [4] (adapted from the general **LK**-case to **LJ**) reducing the mix in ψ. $\Gamma_1, \Gamma_2, \Pi, \Delta$ are multisets of formulas and $|\Delta| \leq 1$.

3.11. rank $= 2$

3.111. $\psi_1 = A \vdash A$:

$$\dfrac{A \vdash A \quad \begin{array}{c}\vdots\,\psi_2\\ \Pi \vdash \Delta\end{array}}{A, \Pi^* \vdash \Delta}\ mix(A)$$

transforms to

$$\dfrac{\begin{array}{c}\vdots\,\psi_2\\ \Pi \vdash \Delta\end{array}}{A, \Pi^* \vdash \Delta}\ c : l^*$$

3.112. $\psi_2 = A \vdash A$: analogous to 3.111

In 3.111 and 3.112 the mix is eliminated completely, i.e. without producing new mixes with lower rank or degree.

3.113.31. The mix-formula is of the form $A \wedge B$

$$\dfrac{\dfrac{\begin{array}{c}\vdots\,\chi_1\\ \Gamma_1 \vdash A\end{array} \quad \begin{array}{c}\vdots\,\chi_2\\ \Gamma_2 \vdash B\end{array}}{\Gamma_1, \Gamma_2 \vdash A \wedge B}\ \wedge : r \qquad \dfrac{\begin{array}{c}\vdots\,\chi_3\\ A, \Gamma_3 \vdash \Delta\end{array}}{A \wedge B, \Gamma_3 \vdash \Delta}\ \wedge : l1}{\Gamma_1, \Gamma_2, \Gamma_3 \vdash \Delta}\ mix(A \wedge B)$$

transforms to

$$\dfrac{\dfrac{\begin{array}{c}\vdots\,\chi_1\\ \Gamma_1 \vdash A\end{array} \quad \begin{array}{c}\vdots\,\chi_3\\ A, \Gamma_3 \vdash \Delta\end{array}}{\Gamma_1, \Gamma_3^* \vdash \Delta}\ mix(A)}{\Gamma_1, \Gamma_2, \Gamma_3 \vdash \Delta}\ w : l^*$$

(symmetric for $\wedge : l2$)

3.113.33. The mix-formula is of the form $(\forall x)B$

$$\dfrac{\dfrac{\begin{matrix}\vdots\ \chi_1(\alpha)\\ \Gamma_1 \vdash B\{x \leftarrow \alpha\}\end{matrix}}{\Gamma_1 \vdash (\forall x)B}\ \forall : r \qquad \dfrac{\begin{matrix}\vdots\ \chi_2\\ B\{x \leftarrow t\}, \Gamma_2 \vdash \Delta\end{matrix}}{(\forall x)B, \Gamma_2 \vdash \Delta}\ \forall : l}{\Gamma_1, \Gamma_2 \vdash \Delta}\ mix((\forall x)B)$$

transforms to

$$\dfrac{\dfrac{\begin{matrix}\vdots\ \chi_1(t)\\ \Gamma_1 \vdash B\{x \leftarrow t\}\end{matrix} \quad \begin{matrix}\vdots\ \chi_2\\ B\{x \leftarrow t\}, \Gamma_2 \vdash \Delta\end{matrix}}{\Gamma_1, \Gamma_2^* \vdash \Delta}\ mix(B\{x \leftarrow t\})}{\Gamma_1, \Gamma_2 \vdash \Delta}\ w : l^*$$

Mix-formulas of the forms $(\exists x)B, B \vee C, \ldots$ are treated in a similar way.

3.12. rank > 2

3.121. right-rank > 1

3.121.232 The last rule in ψ_2 is $\vee : l$. Then ψ is of the form

$$\dfrac{\begin{matrix}\vdots\ \psi_1\\ \Pi \vdash A\end{matrix} \quad \dfrac{\begin{matrix}\vdots\ \chi_1\\ B, \Gamma_1 \vdash \Delta\end{matrix} \quad \begin{matrix}\vdots\ \chi_2\\ C, \Gamma_2 \vdash \Delta\end{matrix}}{B \vee C, \Gamma_1, \Gamma_2 \vdash \Delta}\ \vee : l}{\Pi, (B \vee C)^*, \Gamma_1^*, \Gamma_2^* \vdash \Delta}\ mix(A)$$

$(B \vee C)^*$ is defined empty if $A = B \vee C$ and $B \vee C$ otherwise. We first define the proof τ:

$$\dfrac{\dfrac{\dfrac{\begin{matrix}\vdots\ \psi_1\\ \Pi \vdash A\end{matrix} \quad \begin{matrix}\vdots\ \chi_1\\ B, \Gamma_1 \vdash \Delta\end{matrix}}{\Pi, B^*, \Gamma_1^* \vdash \Delta}\ mix(A)}{\Pi, B, \Gamma_1^* \vdash \Delta}\ \xi \qquad \dfrac{\dfrac{\begin{matrix}\vdots\ \psi_1\\ \Pi \vdash A\end{matrix} \quad \begin{matrix}\vdots\ \chi_2\\ C, \Gamma_2 \vdash \Delta\end{matrix}}{\Pi, C^*, \Gamma_2^* \vdash \Delta}\ mix(A)}{\Pi, C, \Gamma_2^* \vdash \Delta}\ \xi}{\Pi, \Pi, B \vee C, \Gamma_1^*, \Gamma_2^* \vdash \Delta}\ \vee : l$$

Note that, in case $A = B$ or $A = C$, the inference ξ is $w : l$; otherwise ξ is the identical transformation and can be dropped. If $(B \vee C)^* = B \vee C$ then ψ transforms to

$$\dfrac{\begin{matrix}\vdots\ \tau\\ \Pi, \Pi, B \vee C, \Gamma_1^*, \Gamma_2^* \vdash \Delta\end{matrix}}{\Pi, B \vee C, \Gamma_1^*, \Gamma_2^* \vdash \Delta}\ c : l^*$$

If, on the other hand, $(B \vee C)^*$ is empty (i.e. $B \vee C = A$) then we transform

ψ to

$$\dfrac{\dfrac{\begin{array}{c}\vdots\ \psi_1\\ \Pi \vdash A\end{array} \quad \begin{array}{c}\vdots\ \tau\\ \Pi, \Pi, A, \Gamma_1^*, \Gamma_2^* \vdash \Delta\end{array}}{\Pi, \Pi^*, \Pi^*, \Gamma_1^*, \Gamma_2^* \vdash \Delta}\ mix(A)}{\Pi, \Gamma_1^*, \Gamma_2^* \vdash \Delta}\ c : l^*$$

□

Corollary 4.1 (Subformula Property). **LJ** has the *subformula property*: For every provable sequent S there exists a proof that contains only subformulas (modulo terms) of formulas from S.

Proof. By induction on the structure of cut-free proofs (by Theorem 4.1 it suffices to consider cut-free proofs only). □

Corollary 4.2 (Consistency). **LJ** is consistent.

Proof. Assume **LJ** is not consistent, i.e. a sequent of the form $\vdash A \wedge \neg A$ has a proof, say χ. Then there would exist a proof of the empty sequent $\vdash$ as follows:

$$\dfrac{\begin{array}{c}\vdots\ \chi\\ \vdash A \wedge \neg A\end{array} \quad \dfrac{\dfrac{A \vdash A}{A, \neg A \vdash}\ \neg : l}{A \wedge \neg A \vdash}\ \wedge : l1, \wedge : l2, c : l}{\vdash}\ cut$$

Then, by Theorem 4.1 we could create a cut-free proof of $\vdash$ that (due to the subformula property) would contain only subformulas of the empty sequent which is clearly impossible. □

Corollary 4.3. **LJ** provides a decision procedure for propositional intuitionistic logic.

Proof Sketch. A reduced sequent is a sequent that contains each formula in both the antecedens and in the consequent at most three times. As Gentzen showed in [7] every proof of a reduced end-sequent can be modified to contain reduced sequents only. If we want to decide the derivability of a propositional sequent S we take its reduced equivalent S' and create all sequents that contain only subformulas of S'. These finitely many sequents are the only sequents we can use to build a proof of S'. □

Note that the proof that $\vdash A \vee \neg A$ is not derivable in **LJ** outlined in example 2.3 also depends on this theorem as it does not talk about the cut-rule.

Theorem 4.1 as well as the corollaries 4.1, 4.2 and 4.3 also hold for the more general case of **LK**, the classical counterpart of **LJ**.

4.2 Normalization

The counterpart of cut-elimination in the sequent calculus is normalization in natural deduction (and in the typed lambda calculus). Although there exists no explicit cut rule in **NJ** there are combinations of rules that correspond to cuts: an introduction rule followed immediately by an elimination rule on the same formula. Due to the Curry-Howard isomorphism we can treat normalization in natural deduction and in the typed lambda calculus as the same relation. We will switch between these two formalisms as appropriate.

Definition 4.4 (Notions of Reduction). A *notion of reduction* in the typed lambda calculus is a binary relation $R \subseteq \Lambda_{\mathbb{T}} \times \Lambda_{\mathbb{T}}$.

We will make use of the following notions of reduction:

1. $\beta F = \{((\lambda \mathsf{x}^A.\mathsf{M})\mathsf{N}, \mathsf{M}\{\mathsf{x}^A \leftarrow \mathsf{N}\}) \mid A \in \mathrm{PL}^-, \mathsf{M}, \mathsf{N} \in \Lambda_{\mathbb{T}}, \mathsf{N} : A\}$
2. $\beta i = \{((\lambda \mathsf{x}^i.\mathsf{M})t, \mathsf{M}\{\mathsf{x}^i \leftarrow t\}) \mid \mathsf{M} \in \Lambda_{\mathbb{T}}, t : i\}$
3. $\pi = \{(\pi^1\langle \mathsf{M}, \mathsf{N}\rangle, \mathsf{M}) \mid \mathsf{M}, \mathsf{N} \in \Lambda_{\mathbb{T}}\} \cup \{(\pi^2\langle \mathsf{M}, \mathsf{N}\rangle, \mathsf{N}) \mid \mathsf{M}, \mathsf{N} \in \Lambda_{\mathbb{T}}\}$
4. $\beta = \beta F \cup \beta i$

In the above notions of reductions the first component of each element is called *redex* and the second component is called *contractum*. In natural deduction, these notions of reduction correspond to the following proof transformations (the left side being the redex and the right side being the contractum):

βF:

$$\cfrac{\begin{array}{c}\vdots\ \varphi_1\\ A\end{array} \qquad \cfrac{\begin{array}{c}[\![A]\!]^l\\ \vdots\ \varphi_2\\ B\end{array}}{A \to B}\to I^l}{B}\to E \qquad\Longrightarrow\qquad \begin{array}{c}\vdots\ \varphi_1\\ A\\ \vdots\ \varphi_2\\ B\end{array}$$

This means that φ_1 is inserted for *each* assumption $[\![A]\!]^l$.

βi:

$$\cfrac{\cfrac{\begin{array}{c}\vdots\ \varphi(\alpha)\\ A\{x \leftarrow \alpha\}\end{array}}{(\forall x)A}\forall I}{A\{x \leftarrow t\}}\forall E \qquad\Longrightarrow\qquad \begin{array}{c}\vdots\ \varphi(t)\\ A\{x \leftarrow t\}\end{array}$$

π:

$$\dfrac{\dfrac{\begin{matrix}\vdots\ \varphi_1 & \vdots\ \varphi_2 \\ A & B\end{matrix}}{A \wedge B}\wedge I}{A}\wedge E1 \quad \Longrightarrow \quad \begin{matrix}\vdots\ \varphi_1 \\ A\end{matrix}$$

symmetrical for $\wedge E2$

The proof figures on the left are sometimes also called $\to$-,$\forall$-,$\wedge$-*cuts in* **NJ** and the transformations are called $\to$-,$\forall$-,$\wedge$-contractions.

Definition 4.5 (Reduction Relations). For $R \in \{\beta F, \beta i, \pi, \beta\}$ we define the following reduction relations on $\Lambda_{\mathbb{T}}$:

1. The compatible closure $\to_R$
   (a) $(\mathsf{M}, \mathsf{N}) \in R \Rightarrow \mathsf{M} \to_R \mathsf{N}$
   (b) $\mathsf{M} \to_R \mathsf{N} \Rightarrow \mathsf{ZM} \to_R \mathsf{ZN}$
   (c) $\mathsf{M} \to_R \mathsf{N} \Rightarrow \mathsf{MZ} \to_R \mathsf{NZ}$
   (d) $\mathsf{M} \to_R \mathsf{N} \Rightarrow \lambda\mathsf{x}.\mathsf{M} \to_R \lambda\mathsf{x}.\mathsf{N}$
   (e) $\mathsf{M} \to_R \mathsf{N} \Rightarrow \langle \mathsf{M}, \mathsf{Z} \rangle \to_R \langle \mathsf{N}, \mathsf{Z} \rangle$
   (f) $\mathsf{M} \to_R \mathsf{N} \Rightarrow \langle \mathsf{Z}, \mathsf{M} \rangle \to_R \langle \mathsf{Z}, \mathsf{N} \rangle$
   (g) $\mathsf{M} \to_R \mathsf{N} \Rightarrow \pi^1\mathsf{M} \to_R \pi^1\mathsf{N}$
   (h) $\mathsf{M} \to_R \mathsf{N} \Rightarrow \pi^2\mathsf{M} \to_R \pi^2\mathsf{N}$
2. The reflexive and transitive closure $\twoheadrightarrow_R$ of $\to_R$
   (a) $\mathsf{M} \to_R \mathsf{N} \Rightarrow \mathsf{M} \twoheadrightarrow_R \mathsf{N}$
   (b) $\mathsf{M} \twoheadrightarrow_R \mathsf{M}$
   (c) $\mathsf{M} \twoheadrightarrow_R \mathsf{N}, \mathsf{N} \twoheadrightarrow_R \mathsf{L} \Rightarrow \mathsf{M} \twoheadrightarrow_R \mathsf{L}$
3. The equivalence relation $=_R$ generated by $\twoheadrightarrow_R$
   (a) $\mathsf{M} \twoheadrightarrow_R \mathsf{N} \Rightarrow \mathsf{M} =_R \mathsf{N}$
   (b) $\mathsf{M} =_R \mathsf{N} \Rightarrow \mathsf{N} =_R \mathsf{M}$
   (c) $\mathsf{M} =_R \mathsf{N}, \mathsf{N} =_R \mathsf{L} \Rightarrow \mathsf{M} =_R \mathsf{L}$

Note that for $\mathsf{M} : A$ and $\mathsf{M} \to_R \mathsf{N}$ also $\mathsf{N} : A$. The same holds also for $\twoheadrightarrow_R$ and $=_R$. In **NJ** this means that the proven formula is not changed, so $\to_R$ and $\twoheadrightarrow_R$ can be regarded as proof transformations and $=_R$ as proof equivalence modulo normalization.

Definition 4.6 (Normal Form). Let $\mathsf{M} \in \Lambda_{\mathbb{T}}$. For $R \in \{\beta F, \beta i, \pi, \beta\}$, M is in *R-normal form* if there is no $\mathsf{N} \in \Lambda_{\mathbb{T}}$ such that $\mathsf{M} \to_R \mathsf{N}$.

A somewhat simpler (but equivalent) characterization of normal forms is to check wether $\mathsf{M} \in \Lambda_{\mathbb{T}}$ has a subterm of the form of a redex of one of the notions of reduction (see Definition 4.4). If not, then M is in the respective normal form.

4.2.1 Properties of Normalization

In this section, we will sketch the proofs of two important properties of these reductions. The first being the so called *Church-Rosser property* that guarantees the uniqueness of a normal form (if it exists) and the second the *strong normalization property* that guarantees the existence of a normal form for all terms. The Church-Rosser property does even hold for type-free calculus, the strong normalization property does not.

Definition 4.7 (Diamond Property[1]). A reduction relation $\rightarrow$ satisfies the *diamond property* if for all $\mathsf{M}, \mathsf{M}_1, \mathsf{M}_2 \in \Lambda_{\mathbb{T}}$ where $\mathsf{M} \rightarrow \mathsf{M}_1$ and $\mathsf{M} \rightarrow \mathsf{M}_2$ there exists an $\mathsf{N} \in \Lambda_{\mathbb{T}}$ such that $\mathsf{M}_1 \rightarrow \mathsf{N}$ and $\mathsf{M}_2 \rightarrow \mathsf{N}$.

Definition 4.8 (Church-Rosser Property). A notion of reduction R satisfies the *Church-Rosser property* (CR) if $\twoheadrightarrow_R$ satisfies the diamond property.

Lemma 4.1. Let $\rightarrow$ be a binary relation and let $\rightarrow^*$ be its transitive closure. Then, if $\rightarrow$ satisfies the diamond property, also $\rightarrow^*$ satisfies the diamond property.

Proof Sketch. By induction on the number of $\rightarrow$-reduction steps that one $\rightarrow^*$-reduction step contains. □

Theorem 4.2. β is CR.

Proof Sketch. By defining a binary relation $\twoheadrightarrow_1$ that is CR and contains the reflexive closure of $\rightarrow_\beta$. The reduction relation $\twoheadrightarrow_1$ is compatible but not transitive, its transitive closure is $\twoheadrightarrow_\beta$. Applying Lemma 4.1 proves the theorem (a detailed proof can be found in [5]). □

Theorem 4.3. π is CR.

Proof Sketch. By induction on the term structure. □

Definition 4.9. Let $\rightarrow_1$ and $\rightarrow_2$ be two binary relations on a set X. Then $\rightarrow_1$ and $\rightarrow_2$ *commute* if for all $x, x_1, x_2 \in X$ such that $x \rightarrow_1 x_1$ and $x \rightarrow_2 x_2$ there exists a $y \in X$ such that $x_1 \rightarrow_2 y$ and $x_2 \rightarrow_1 y$.

Lemma 4.2. $\twoheadrightarrow_\beta$ commutes with $\twoheadrightarrow_\pi$

Proof Sketch. By induction on the number of reduction steps. □

Lemma 4.3 (Hindley-Rosen). Let R_1 and R_2 be two notions of reduction. If R_1 and R_2 are CR and $\twoheadrightarrow_{R_1}$ commutes with $\twoheadrightarrow_{R_2}$ then $R_1 \cup R_2$ is CR.

[1] In the literature this property is sometimes also called *confluence*.

Proof Sketch. By induction on the number of reduction steps. □

Theorem 4.4 (Church-Rosser Property). $\beta \cup \pi$ is CR.

Proof. By Theorems 4.2 and 4.3 and Lemmas 4.2 and 4.3. □

We know now that the normal form of a term is unique if it exists. However, it may still be possible that some terms do not have a normal form at all. Indeed, this is the case in the type-free lambda calculus but it is not the case in the typed lambda calculus. In the following we will sketch the proof of the strong normalization theorem from [12].

Definition 4.10 (Strong Normalization). A term $\mathsf{M} \in \Lambda_{\mathbb{T}}$ is called *strongly normalizing* (SN) iff there is no infinite sequence of terms $\mathsf{M_1}, \mathsf{M_2}, \ldots$ with $\mathsf{M_i} \to \mathsf{M_{i+1}}$.

Lemma 4.4. Let A be a PL$^-$-formula. Let $\mathsf{P} : A$ be a term of the form

$$\text{(i) } \mathsf{P} \equiv \mathsf{MN} \qquad \text{(ii) } \mathsf{P} \equiv (\lambda \mathsf{x}.\mathsf{M})\mathsf{N}$$
$$\text{(iii) } \mathsf{P} \equiv \langle \mathsf{M}, \mathsf{N} \rangle \qquad \text{(iv) } \mathsf{P} \equiv \pi_1 \mathsf{M} \qquad \text{(v) } \mathsf{P} \equiv \pi_2 \mathsf{M}$$

Then P is SN if M and N are SN.

Proof Sketch. By induction on the structure of A. □

Lemma 4.5. Let M be a term whose free variables are $\mathsf{x_1} : A_1, \ldots, \mathsf{x_n} : A_n$. If $\mathsf{N_1} : A_1, \ldots, \mathsf{N_n} : A_n$ are SN then also the term $\mathsf{M}\{\mathsf{x_1} \leftarrow \mathsf{N_1}\} \ldots \{\mathsf{x_n} \leftarrow \mathsf{N_n}\}$ is SN.

Proof Sketch. By induction on the structure of M using Lemma 4.4. □

Theorem 4.5 (Strong Normalization). For each PL$^-$-formula A every term $\mathsf{M} \in \Lambda_A$ is strongly normalizing.

Proof Sketch. By Lemma 4.5. □

Putting the Church-Rosser and strong normalization properties together, one obtains the nice property that each term has a unique normal form.

Corollary 4.4. The equivalence modulo normalization $=_{\beta \cup \pi}$ of typed lambda terms (respectively **NJ**-proofs) is decidable.

Proof. To decide wether $\mathsf{M} =_{\beta \cup \pi} \mathsf{N}$ compute the $\beta \cup \pi$ normal forms M' and N' and check wether $\mathsf{M}' \equiv \mathsf{N}'$. By Theorem 4.5 the terms M' and N' can be computed and by Theorem 4.4 it holds that $\mathsf{M}' \equiv \mathsf{N}'$ iff $\mathsf{M} =_{\beta \cup \pi} \mathsf{N}$. □

4.2.2 Normalization and the Curry-Howard Isomorphism

Due to the Curry-Howard isomorphism, normalization in the formalism of natural deduction behaves the same way as in the typed lambda calculus.

Example 4.1. This example illustrates both the Curry-Howard isomorphism and the reductions in the typed lambda calculus (or **NJ** respectively). Consider the following **NJ**-proof of $P(f(a)) \land Q(f(a))$ from (the active assumptions) $(\forall x)P(x)$ and $(\forall y)Q(y)$ (printed in **boldface**). It contains an $\to$-cut (βF-redex).

$$\dfrac{\dfrac{\dfrac{(\forall \mathbf{x})\mathbf{P}(\mathbf{x}) \quad (\forall \mathbf{y})\mathbf{Q}(\mathbf{y})}{(\forall x)P(x) \land (\forall y)Q(y)}\land I \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[\![(\forall x)P(x) \land (\forall y)Q(y)]\!]^1}{(\forall x)P(x)}\land E1}{P(\alpha)}\forall E \quad \dfrac{\dfrac{[\![(\forall x)P(x) \land (\forall y)Q(y)]\!]^1}{(\forall y)Q(y)}\land E2}{Q(\alpha)}\forall E}{P(\alpha) \land Q(\alpha)}\land I}{(\forall z)(P(z) \land Q(z))}\forall I}{((\forall x)P(x) \land (\forall y)Q(y)) \to (\forall z)(P(z) \land Q(z))}\to I^1}{(\forall z)(P(z) \land Q(z))}\to E}{P(f(a)) \land Q(f(a))}\forall E$$

The corresponding lambda term is

$$(\lambda \mathsf{x}_1.\lambda \mathsf{z}.\langle(\pi^1\mathsf{x}_1)\mathsf{z}, (\pi^2\mathsf{x}_1)\mathsf{z}\rangle)\langle \mathsf{u}, \mathsf{v}\rangle f(a) : (P(f(a)) \land Q(f(a)))$$

with

$$\mathsf{u} : (\forall x)P(x), \mathsf{v} : (\forall y)Q(y), \mathsf{x}_1 : (\forall x)P(x) \land (\forall y)Q(y), \alpha : i, \mathsf{z} : i$$

Applying βF-reduction (corresponding to $\to$-contraction of the proof) yields

$$\to_{\beta F} (\lambda \mathsf{z}.\langle(\pi^1\langle \mathsf{u}, \mathsf{v}\rangle)\mathsf{z}, (\pi^2\langle \mathsf{u}, \mathsf{v}\rangle)\mathsf{z}\rangle) f(a) : P(f(a)) \land Q(f(a))$$

which corresponds to the proof

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(\forall \mathbf{x})\mathbf{P}(\mathbf{x}) \quad (\forall \mathbf{y})\mathbf{Q}(\mathbf{y})}{(\forall x)P(x) \land (\forall y)Q(y)}\land I}{(\forall x)P(x)}\land E1}{P(\alpha)}\forall E \quad \dfrac{\dfrac{\dfrac{(\forall \mathbf{x})\mathbf{P}(\mathbf{x}) \quad (\forall \mathbf{y})\mathbf{Q}(\mathbf{y})}{(\forall x)P(x) \land (\forall y)Q(y)}\land I}{(\forall y)Q(y)}\land E2}{Q(\alpha)}\forall E}{P(\alpha) \land Q(\alpha)}\land I}{(\forall z)(P(z) \land Q(z))}\forall I}{P(f(a)) \land Q(f(a))}\forall E$$

While this lambda term is now in βF normal form, it is no longer in βi- nor in π-normal form. A new $\forall$-cut and two new $\land$-cuts have been introduced by the above βF-reduction. By the Church-Rosser property of $\beta \cup \pi$ it does not matter which reduction we choose next. By applying βi-reduction we obtain:

$$\langle(\pi^1\langle\mathsf{u},\mathsf{v}\rangle)f(a),(\pi^2\langle\mathsf{u},\mathsf{v}\rangle)f(a)\rangle : P(f(a))\wedge Q(f(a))$$

which corresponds to the proof

$$\dfrac{\dfrac{\dfrac{\dfrac{(\forall\mathbf{x})\mathbf{P}(\mathbf{x})\quad(\forall\mathbf{y})\mathbf{Q}(\mathbf{y})}{(\forall x)P(x)\wedge(\forall y)Q(y)}\wedge I}{(\forall x)P(x)}\wedge E1}{P(f(a))}\forall E \qquad \dfrac{\dfrac{\dfrac{(\forall\mathbf{x})\mathbf{P}(\mathbf{x})\quad(\forall\mathbf{y})\mathbf{Q}(\mathbf{y})}{(\forall x)P(x)\wedge(\forall y)Q(y)}\wedge I}{(\forall y)Q(y)}\wedge E2}{Q(f(a))}\forall E}{P(f(a))\wedge Q(f(a))}\wedge I$$

Applying π-reduction twice yields

$$\langle(\pi^1\langle\mathsf{u},\mathsf{v}\rangle)f(a),(\pi^2\langle\mathsf{u},\mathsf{v}\rangle)f(a)\rangle \to_\pi \langle\mathsf{u}f(a),(\pi^2\langle\mathsf{u},\mathsf{v}\rangle)f(a)\rangle \to_\pi \langle\mathsf{u}f(a),\mathsf{v}f(a)\rangle$$

which is in $\beta\cup\pi$ normal form and corresponds to the proof

$$\dfrac{\dfrac{(\forall\mathbf{x})\mathbf{P}(\mathbf{x})}{P(f(a))}\forall E \qquad \dfrac{(\forall\mathbf{y})\mathbf{Q}(\mathbf{y})}{Q(f(a))}\forall E}{P(f(a))\wedge Q(f(a))}\wedge I$$

Chapter 5

Cut-Projection

In this chapter we will describe the method of cut projection (presented in [1] and [2]) which differs substantially from Gentzen's method. The characteristic feature of Gentzen's cut elimination "algorithm" is the double induction on rank and degree. The rank of a cut is reduced (by shifting it upwards) until it reaches 2. Only after that the degree of the cut (the logical complexity of the cut formula) is reduced. The method of Cut-Projection on the other hand works by leaving the cuts in place but reducing the degree nonetheless. This leads to a method which reduces the cuts without increasing the size of the proof. It is clear that cut-projection can non eliminate all cuts because if it could it would be a linear cut-elimination algorithm which is impossible because the cut-elimination problem is of non-elementary complexity.

Example 5.1. Consider this **LJ**-proof:

$$
\begin{prooftree}
\AxiomC{$Q(\alpha) \vdash Q(\alpha)$}
\RightLabel{$w : l$}
\UnaryInfC{$Q(\alpha), P(\alpha) \vdash Q(\alpha)$}
\RightLabel{$\rightarrow : r$}
\UnaryInfC{$Q(\alpha) \vdash P(\alpha) \rightarrow Q(\alpha)$}
\RightLabel{$\forall : l$}
\UnaryInfC{$(\forall y)Q(y) \vdash P(\alpha) \rightarrow Q(\alpha)$}
\RightLabel{$\forall : r$}
\UnaryInfC{$(\forall y)Q(y) \vdash (\forall x)(P(x) \rightarrow Q(x))$}
\RightLabel{$\wedge : l2$}
\UnaryInfC{$(\forall x)P(x) \wedge (\forall y)Q(y) \vdash (\forall x)(P(x) \rightarrow Q(x))$}
\AxiomC{$P(a) \vdash P(a)$}
\AxiomC{$Q(a) \vdash Q(a)$}
\RightLabel{$\rightarrow : l$}
\BinaryInfC{$P(a) \rightarrow Q(a), P(a) \vdash Q(a)$}
\RightLabel{$\forall : l$}
\UnaryInfC{$(\forall x)(P(x) \rightarrow Q(x)), P(a) \vdash Q(a)$}
\RightLabel{$\rightarrow : r$}
\UnaryInfC{$(\forall x)(P(x) \rightarrow Q(x)) \vdash P(a) \rightarrow Q(a)$}
\RightLabel{cut}
\BinaryInfC{$(\forall x)P(x) \wedge (\forall y)Q(y) \vdash P(a) \rightarrow Q(a)$}
\end{prooftree}
$$

The universal quantifier in the cut formula is too general in the sense that the only x it talks about (in this proof) is a. The cut formula thus can be simplified by replacing x by a and omitting the quantifier. This also reduces the size of the proof. The result is:

$$
\cfrac{\cfrac{\cfrac{\cfrac{Q(a) \vdash Q(a)}{Q(a), P(a) \vdash Q(a)}\ w:l}{Q(a) \vdash P(a) \to Q(a)}\ \to: r}{(\forall y)Q(y) \vdash P(a) \to Q(a)}\ \forall : l}{(\forall x)P(x) \wedge (\forall y)Q(y) \vdash P(a) \to Q(a))}\ \wedge : l2 \quad \cfrac{\cfrac{P(a) \vdash P(a) \quad Q(a) \vdash Q(a)}{P(a) \to Q(a), P(a) \vdash Q(a)}\ \to: l}{P(a) \to Q(a) \vdash P(a) \to Q(a)}\ \to: r
$$
$$
\overline{(\forall x)P(x) \wedge (\forall y)Q(y) \vdash P(a) \to Q(a)}\ cut
$$

Note that it is essential for this transformation that there is only *one single* term that is substituted for x on the right side. If the right side would be a proof that needs another instantiation of the cut formula with another term b then it would not be clear if the new cut formula should be $P(a) \to Q(a)$ or $P(b) \to Q(b)$.

The method of Cut-Projection is the above proof transformation generalized to subformulas of the cut-formula having one of $\vee, \wedge, \exists, \forall$ as main logical symbol: The essence of cut projection is to reduce the logical complexity of a cut formula A by choosing a subformula B of A and a formula B^* (that is basically a subformula of B) and replacing B by B^* in the proof above A. In the example above we have $A = (\forall x)(P(x) \to Q(x)), B = A$ and $B^* = P(a) \to Q(a)$.

In Section 5.1 we will describe the method of Cut-Projection (presented in [1]) for the sequent calculus, in section 5.2 we will show how it can be used for cut elimination in a certain subclass of proofs (see [2]) and in section 5.3 we will also show how this method can be applied to proofs in natural deduction.

5.1 Projection-Based Cut-Elimination

For defining cut projection an important notion is that of a predecessor in an **LJ**-proof.

Definition 5.1 (Direct Predecessor). Let χ be an **LJ**-proof and θ be an occurrence of a formula A in χ. An occurrence τ of a formula B is a *direct predecessor* of θ ($\tau \lhd_\chi \theta$) if τ and θ are connected by a rule application ξ and

1. If $|\theta|$ is the main formula of ξ and $|\tau|$ is an auxiliary formula of ξ
2. If $|\theta|$ is not the main formula of ξ (then $A = B$) and τ is the formula occurrence corresponding to θ.

Note that, if $|\theta|$ is the main formula of a weakening rule then there exists no τ with $\tau \lhd \theta$. The notion of the "corresponding formula" in the above definition has some ambiguity in a multiset sequent structure. Consider for example the following rule application with two occurrences of the formula A in each sequent[1]:

$$\frac{\Gamma, A^{\tau}, A^{\gamma} \vdash B}{\Gamma, A^{\theta}, A^{\lambda} \vdash B \vee C} \vee : r1$$

It is not clear how to answer the question which formula occurrence is the predecessor of A^{θ}, it could be both A^{τ} and A^{γ}. A natural way to solve such a situation is to let the i-th occurrence of A in the premise sequent be the predecessor of the i-th occurrence of A in the conclusion sequent. But in a multiset there is no ordering of the formulas, so the notion of i-th occurrence (of a formula) is not definable. These difficulties could be solved by adopting Gentzen's original sequent structure as two *sequences* of formulas or by indexing formulas. But to avoid a too complicated notation, we will in the following assume a unique notion of "corresponding formula" in this sense.

As formal foundation of cut projection the predecessor relation on formula occurrences is not enough. We also want to be able to talk about predecessors of subformula occurrences (in a proof).

Definition 5.2 (Subformula Predecessor). The direct subformula predecessor relation $<_{\chi}$ in an **LJ**-proof χ is defined as follows. Let λ be a subformula occurrence in χ and let ξ be the rule application above λ.

1. λ is a subformula occurrence in the main formula of ξ
   (a) If $|\lambda|$ is the main formula of a logical rule or if λ occurs in the main formula of a weakening rule or in the disappearing part of the main formula of an $\wedge : l$- or $\vee : r$-rule, then it has no predecessor.
   (b) If λ occurs in the main formula of a logical rule (but $|\lambda|$ is not *equal* to the main formula) then $\lambda_1 <_{\chi} \lambda$ (and $\lambda_2 <_{\chi} \lambda$) where λ_1 (and λ_2) is (are) the corresponding subformula(s) in the auxiliary formula(s).
2. λ is not a subformula occurrence in the main formula of ξ
   Then λ is not modified by ξ and $\lambda' <_{\chi} \lambda$ for the corresponding (in the above sense) subformula occurrence λ' in the sequent above ξ.

The subformula predecessor relation $\ll_{\chi}$ is the transitive closure of $<_{\chi}$.

We will treat subformula occurrences in a proof connected by the relations $<_{\chi}$ and $\ll_{\chi}$ as objects, more precisely as *predecessor trees*. For a subformula occurrence μ in an **LJ**-proof χ the predecessor tree $\mathcal{T}_{\mu}$ is defined as follows:

[1] We use the notation A^{θ} to denote the formula occurrence θ of the formula A.

1. The root of $\mathcal{T}_\mu$ is μ
2. λ' is a child of λ in $\mathcal{T}_\mu$ iff $\lambda' <_\chi \lambda$.

We will sometimes also write $\mathcal{T}$ instead of $\mathcal{T}_\mu$ if it is clear from the context which subformula occurrence is the root of the predecessor tree. A leaf of a predecessor tree is a node that has no children. A leaf λ of a predecessor tree is called *open* if λ is an occurrence of a subformula in an axiom. A leaf is *closed* iff it is not open. A predecessor tree is called closed iff all leaves are closed.

Note that in an **LJ**-proof χ with only atomic axioms all leaves of all predecessor trees of all non-atomic subformulas occurrences are closed. For a leaf to be open there would have to exist a non-atomic subformula in an atomic axiom sequent which clearly is impossible. To simplify notation we will assume that we are only dealing with closed predecessor trees[2].

There is an important distinction between two different kinds of leaves. In Definition 5.2, Case 1a there are two different reasons why a subformula occurrence λ has no predecessor (i.e. is a leaf): The first reason is that λ is the main formula of a logical rule and thus is decomposed by this rule, this will be called *main-symbol closure*. The second reason is that λ occurs in the main formula of a weakening rule or in the disappearing part of an $\wedge : l$- or $\wedge : r$-rule, this will be called *weakening-like closure*.

Note that for two subformula occurrences μ and λ with $\lambda \ll_\chi \mu$ there exists a substitution σ such that $|\mu|\sigma = |\lambda|$. We will now describe these substitutions in more detail.

Definition 5.3 (Associated Substitution). The substitution σ_λ associated with a node λ in a predecessor tree $\mathcal{T}_\mu$ is defined as follows:

1. If $\lambda = \mu$ then $\sigma_\lambda = id$
2. If λ has a parent λ' connected via a rule ξ (so λ is above ξ, λ' is below ξ): If $\xi \in \{\forall : l, \forall : r, \exists : l, \exists : r\}$ and λ' is in the scope of the quantifier removed by ξ then $\sigma_\lambda = \{x \leftarrow t\}\sigma_{\lambda'}$ for the term t introduced by ξ (Note that for $\xi = \forall : r$ and $\xi = \exists : l$ the term t is an eigenvariable).

In any other case $\sigma_\lambda = \sigma_{\lambda'}$.

The associated substitutions fulfill the property mentioned above: For a subformula occurrence μ and a subformula occurrence λ in $\mathcal{T}_\mu$: $|\mu|\sigma_\lambda = |\lambda|$.

Definition 5.4 (Basic Projection Targets). Let B be a formula with one of $\vee, \wedge, \exists, \forall$ as main logical symbol. The set of basic projection targets $\mathcal{B}_B$ is defined as follows:

[2] An open predecessor tree can easily be modified to be closed by proving all axioms of the form $F \vdash F$ (for an arbitrary formula F) with atomic axioms.

1. If $B = C \vee D$ or $B = C \wedge D$ then $\mathcal{B}_B = \{C, D\}$
2. If $B = (\exists x)C$ or $B = (\forall x)C$ then $\mathcal{B}_B = \{C\{x \leftarrow t\} \mid t \in T\}$.

These are the basic projection targets in the sense that each subformula B of a cut formula A must be projected to some formula $B^* \in \mathcal{B}_B$ if it is projected. However, projection is not always possible and, if it is, we can not choose an arbitrary formula $B^* \in \mathcal{B}_B$. We will now establish criteria when projection is permitted and which projection targets are permitted.

Definition 5.5 (Critical Occurrences). Let χ be an **LJ**-proof, A be a cut formula in χ, B be a subformula of A, μ_l the occurrence of B on the left side (of the cut rule) and μ_r on the right side.

1. B is of the form $B = C \vee D$ or $B = (\exists x)C$ then μ_l (μ_r) is called a *critical occurrence* if it is of positive (negative) polarity in A.
2. B is of the form $B = C \wedge D$ or $B = (\forall x)C$ then μ_l (μ_r) is called a *critical occurrence* if it is of negative (positive) polarity in A:

For projection of a subformula B of a cut formula to be permitted, the following two conditions have to hold:

1. the *uniqueness condition* (UC):
   The uniqueness condition is essential for cut projection: Let μ be the critical occurrence of B, let $\xi_1, \ldots, \xi_n$ be the rule applications above the main-symbol closure leaves of $\mathcal{T}_\mu$ and let $\lambda_1, \ldots, \lambda_n$ be the occurrences of the auxiliary formulas of $\xi_1, \ldots, \xi_n$ (there is only one auxiliary formula for each ξ_i because μ is critical). Then μ fulfills the uniqueness condition (UC) if $|\lambda_1|\sigma_{\lambda_1} = |\lambda_2|\sigma_{\lambda_2} = \ldots = |\lambda_n|\sigma_{\lambda_n}$.
2. the *eigenvariable condition*:
   For $B = (\exists x)C$ or $B = (\forall x)C$ the basic targets are of the form $C\{x \leftarrow t\}$. Only those targets are allowed where insertion of t does not violate an eigenvariable condition of an $\exists : l$- or $\forall : r$-rule application.

Definition 5.6 (Projection Targets). Let χ be an **LJ**-proof, A be a cut-formula in χ, μ be the critical occurrence of a subformula B in A. The set of projection targets $\mathcal{P}_\mu$ of the predecessor tree $\mathcal{T}_\mu$ is defined inductively on the structure of $\mathcal{T}_\mu$. Let λ be a node in $\mathcal{T}_\mu$ and ξ be the rule above λ.

1. λ is a (closed) leaf
   (a) main-symbol closure
   ξ is a logical rule and $|\lambda|$ is its main formula. Then (because μ is critical) $\xi \in \{\vee : r1, \vee : r2, \wedge : l1, \wedge : l2, \exists : r, \forall : l\}$. Let C' be the auxiliary formula of ξ:
   Then $\mathcal{P}_\lambda = \{C \in \mathcal{B}_B \mid C\sigma_\lambda = C'\}$. Note that $|\mathcal{P}_\lambda| = 1$.

(b) weakening-like closure
ξ is a weakening rule and λ occurs in its main formula or $\xi \in \{\vee : r1, \vee : r2, \wedge : l1, \wedge : l2\}$ and λ occurs in the disappearing part of the main formula:
Then $\mathcal{P}_\lambda = \mathcal{B}_B$

2. λ has a child λ' connected by a rule $\xi \in \{\exists : l, \forall : r\}$
Let α be the eigenvariable in ξ. Then $\mathcal{P}_\lambda = \{F \in \mathcal{P}_{\lambda'} \mid \alpha \notin \mathrm{FV}(F\sigma_\lambda)\}$

3. λ has children λ_1 (and λ_2) connected by a rule $\xi \notin \{\exists : l, \forall : r\}$
Then $\mathcal{P}_\lambda = \mathcal{P}_{\lambda_1}(\cap \mathcal{P}_{\lambda_2})$

Projection of a subformula B of a cut formula A is possible iff $\mathcal{P}_\mu \neq \emptyset$ for the critical occurrence μ of B. Then projection of B to a $B^* \in \mathcal{P}_\mu$ is done essentially by replacing all predecessors of B by B^*, formally:

Definition 5.7 (Cut-Projection). Let χ be a proof of the form

$$\dfrac{\begin{matrix}\vdots\ \chi_1 \\ \Gamma \vdash A\end{matrix} \qquad \begin{matrix}\vdots\ \chi_2 \\ A, \Pi \vdash \Delta\end{matrix}}{\Gamma, \Pi \vdash \Delta}\ cut$$

and B be a subformula of A, let μ be the critical and θ be the non-critical occurrence of B in A and let $\mathcal{T}_\mu$ and $\mathcal{T}_\theta$ be the closed predecessor trees of μ and θ.

1. Select a projection target $B^* \in \mathcal{P}_\mu$

2. For all non-leaf nodes λ of $\mathcal{T}_\mu$ and $\mathcal{T}_\theta$: Replace the formula at λ by $B^*\sigma_\lambda$.

3. For all leaves λ of the critical predecessor tree $\mathcal{T}_\mu$ let ψ be the (sub-)proof above λ and ξ be the rule above λ.

   (a) main-symbol closure

      i. $\xi = \vee : r1$ (or $\xi = \vee : r2$)
      Then ψ is of the form

$$\dfrac{\begin{matrix}\vdots\ \varphi \\ \Gamma \vdash C'\end{matrix}}{\Gamma \vdash C' \vee D'}\ \vee : r1$$

      and B is of the form $B = C \vee D$ with $C' = C\sigma_\lambda$ and $D' = D\sigma_\lambda$ and $B^* = C$ (or $B^* = D$). We replace ψ by φ.

ii. $\xi = \exists : r$
Then ψ is of the form

$$\dfrac{\begin{array}{c}\vdots\ \varphi \\ \Gamma \vdash C'\{x \leftarrow t'\}\end{array}}{\Gamma \vdash (\exists x)C'}\ \exists : r$$

B is of the form $B = (\exists x)C$ with $C' = C\sigma_\lambda$ and $B^* = C\{x \leftarrow t\}$ with $t' = t\sigma_\lambda$. We replace ψ by φ.

iii. $\xi = \wedge : l1$ or $\xi = \wedge : l2$: symmetric to Case (i).

iv. $\xi = \forall : l$: symmetric to Case (ii).

(b) weakening-like closure: Replace the formula at λ by $B^*\sigma_\lambda$.

4. For all leaves λ of the non-critical predecessor tree $\mathcal{T}_\theta$. Let ψ be the (sub-)proof above λ and ξ be the rule above λ.

(a) main-symbol closure

i. $\xi = \vee : l$
Then ψ is of the form

$$\dfrac{\begin{array}{c}\vdots\ \varphi_1 \\ C', \Gamma \vdash \Delta\end{array} \quad \begin{array}{c}\vdots\ \varphi_2 \\ D', \Pi, \vdash \Delta\end{array}}{C' \vee D', \Gamma, \Pi \vdash \Delta}\ \vee : l$$

and B is of the form $B = C \vee D$ with $C' = C\sigma_\lambda$ and $D' = D\sigma_\lambda$ and $B^* = C$ (or $B^* = D$). We replace ψ by

$$\dfrac{\begin{array}{c}\vdots\ \varphi_1 \\ C', \Gamma \vdash \Delta\end{array}}{C', \Gamma, \Pi \vdash \Delta}\ w : l^*$$

(or symmetric for $B^* = D$).

ii. $\xi = \exists : l$
Then ψ is of the form

$$\dfrac{\begin{array}{c}\vdots\ \varphi(\alpha) \\ C'\{x \leftarrow \alpha\}, \Gamma \vdash \Delta\end{array}}{(\exists x)C', \Gamma \vdash \Delta}\ \exists : l$$

B is of the form $(\exists x)C$ with $C' = C\sigma_\lambda$ and $B^* = C\{x \leftarrow t\}$ for some term t. Let $t' = t\sigma_\lambda$, we replace ψ by

$$\begin{array}{c}\vdots\ \psi(t') \\ C'\{x \leftarrow t'\}, \Gamma \vdash \Delta\end{array}$$

iii. $\xi = \wedge : r$: symmetric to Case (i).
iv. $\xi = \forall : r$: symmetric to Case (ii).

(b) weakening-like closure: Replace the formula at λ by $B^*\sigma_\lambda$.

Example 5.2. Reconsider the following proof from Example 5.1 (Page 36).

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{Q(\alpha) \vdash Q(\alpha)}{Q(\alpha), P(\alpha) \vdash Q(\alpha)}\, w:l}{Q(\alpha) \vdash P(\alpha) \to Q(\alpha)}\, \to: r}{(\forall y)Q(y) \vdash P(\alpha) \to Q(\alpha)}\, \forall : l}{(\forall y)Q(y) \vdash (\forall x)(P(x) \to Q(x))}\, \forall : r}{(\forall x)P(x) \wedge (\forall y)Q(y) \vdash (\forall x)(P(x) \to Q(x))}\, \wedge : l2 \quad \dfrac{\dfrac{\dfrac{P(a) \vdash P(a) \quad Q(a) \vdash Q(a)}{P(a) \to Q(a), P(a) \vdash Q(a)}\, \to: l}{(\forall x)(P(x) \to Q(x)), P(a) \vdash Q(a)}\, \forall : l}{(\forall x)(P(x) \to Q(x)) \vdash P(a) \to Q(a)}\, \to: r}{(\forall x)P(x) \wedge (\forall y)Q(y) \vdash P(a) \to Q(a)}\, cut$$

The cut formula A is $A = (\forall x)(P(x) \to Q(x))$. The only subformula of A that has one of $\{\wedge, \vee, \forall, \exists\}$ has its top-level symbol is A itself. The critical occurrence of A is that on the right side of the cut, we call it μ. The predecessor tree $\mathcal{T}_\mu$ consists of the two nodes μ and λ where λ is the occurrence of A directly above the $\to: r$-rule. The node λ is a main-symbol closure leaf because $|\lambda|$ is the main formula of the logical rule $\forall : l$. The associated substitutions are $\sigma_\mu = \sigma_\lambda = id$ because there are no quantifier rules on the path from μ to the leaf λ. The set of basic projection targets is $\mathcal{B}_A = \{P(t) \to Q(t) \mid t \in T\}$. The set of projection targets of the predecessor tree $\mathcal{T}_\mu$ is constructed as follows: Let $C' = P(a) \to Q(a)$ be the auxiliary formula of the $\forall : l$-rule, the set of projection targets of λ is $\mathcal{P}_\lambda = \{C \in \mathcal{B}_A \mid C\sigma_\lambda = C'\} = \{P(a) \to Q(a)\}$. And because of $\to: r$ being a unary rule $\mathcal{P}_\mu = \mathcal{P}_\lambda = \{P(a) \to Q(a)\} \neq \emptyset$ so projection of A is possible.

5.2 The Elimination of Monotone Cuts

In this section we will show how the technique of Cut-Projection can be used to eliminate cuts in a syntactic subclass (called $\mathcal{QMON}$) of **LJ**-proofs with only exponential expense. This class includes all Horn theories and in particular all equational theories. The original proof of this result [2] is for **LK**, but $\mathcal{QMON}$ is a subclass of all **LJ**-proofs (which is in turn a subclass of all **LK**-proofs) so this proof can be transfered with minor modifications to **LJ**.

Definition 5.8 (Monotone Formulas). A formula $A \in \mathrm{PL}$ is called *monotone* iff the only logical symbols occurring in A are $\wedge, \vee, \forall, \exists$

Definition 5.9 (Quasi-Monotone Formulas). The set of *quasi-monotone formulas* is defined inductively:

1. Atomic formulas and $\perp$ are quasi-monotone

2. If A and B are quasi-monotone formulas then $(\forall x)A$, $(\exists x)B$ and $A \wedge B$ are quasi-monotone.

3. If A is quasi-monotone and B is monotone then $B \rightarrow A$ is quasi-monotone

A sequent $\Gamma \vdash \Delta$ is called *QM-sequent* iff (all formulas in) Γ are quasi-monotone and (all formulas in) Δ are monotone. In QM-sequents there is no negation and $\vee$ occurs only in positive sequent polarity.

Definition 5.10 ($\mathcal{QMON}$). $\mathcal{QMON}$ is the class of all **LJ**-proofs χ such that

1. the end sequent of χ is a QM-sequent

2. all cut formulas are monotone

Definition 5.11 ($\mathcal{QMON}^*$). A proof χ in $\mathcal{QMON}$ is called *right-normal* if it does not contain weakening to the right. The class of all right-normal proofs will be denoted by $\mathcal{QMON}^*$.

Lemma 5.1 ($\mathcal{QMON}^*$). Let $\chi \in \mathcal{QMON}$ be a proof of a sequent S such that all sequents occurring in χ are QM-sequents. Then there exists a proof $\chi' \in \mathcal{QMON}^*$ of S such that $l(\chi') \leq l(\chi)$.

Proof Sketch. By induction on the structure of $\mathcal{QMON}$-proofs. □

Lemma 5.2 (Uniqueness Condition). Let χ be a proof of a sequent $S : \Gamma \vdash A$ such that $\chi \in \mathcal{QMON}^*$ and let μ be a critical occurrence of a subformula B of A in S. If B is of the form $B = C \vee D$ or $B = (\exists x)C$ then $\mathcal{P}_\mu \neq \emptyset$.

Proof. By the definition of $\mathcal{QMON}$, the absence of the contraction rule on the right and the observation that due to the absence of negation a contraction on the left side can not simulate a contraction on the right side it can be concluded that $\mathcal{P}_\mu$ has only one branch. The lemma follows. □

Lemma 5.3 ($\vee$-Elimination). Let $\chi \in \mathcal{QMON}$ be a proof of the sequent S. Then there exists a proof χ' of S such that $\chi' \in \mathcal{QMON}$, all cut formulas occurring in χ' are $\vee$-free and $l(\chi') \leq l(\chi)$.

Proof. Iterate the following procedure for the proof segment ψ ending with the uppermost, leftmost cut containing $\vee$ in χ until there is no such cut anymore: ψ is of the form:

$$\dfrac{\begin{matrix}\vdots\ \psi_1 \\ \Gamma \vdash A\end{matrix} \qquad \begin{matrix}\vdots\ \psi_2 \\ A, \Pi \vdash \Delta\end{matrix}}{\Gamma, \Pi \vdash \Delta}\ cut$$

Then ψ_1 does only contain QM-sequents (because it does not contain an $\vee$-cut) and by Lemma 5.1 we can replace it by a proof $\psi_1' \in \mathcal{QMON}^*$ of $\Gamma \vdash A$. By Lemma 5.2 for every critical occurrence μ of an $\vee$-subformula in A it holds that $\mathcal{P}_\mu \neq \emptyset$. After applying the cut projection method to each such μ we obtain a new cut formula without $\vee$. □

To eliminate all $\exists$-occurrences from the cut formulas we need to skolemize the proof first. Skolemization is a proof transformation eliminating all strong quantifiers.

Definition 5.12 (Strong and Weak Quantifiers). If $(\forall x)$ occurs positively (negatively) in a formula A then it is called a strong (weak) quantifier. If $(\exists x)$ occurs negatively (positively) in a formula A it is called weak (strong) quantifier.

Definition 5.13 (Skolemization). The function *sk* maps closed formulas into closed formulas:

$sk(F) = F$	if F does not contain strong quantifiers
$= sk(F_{(Qy)}\{y \leftarrow f(x_1, \ldots, x_n)\})$	if (Qy) is in the scope of the weak quantifiers $(Q_1x_1), \ldots, (Q_nx_n)$ (appearing in this order)

where $F_{(Qy)}$ denotes F after omission of (Qy) and f is a new function symbol (or constant symbol if $n = 0$).

Let S be the **LJ**-sequent $A_1, \ldots, A_n \vdash B$ consisting of closed formulas only and let $(A_1' \wedge \ldots \wedge A_n') \rightarrow B' = sk((A_1 \wedge \ldots \wedge A_n) \rightarrow B)$ then $sk(S)$ is defined as $A_1', \ldots, A_n' \vdash B'$.

Lemma 5.4 (Skolemization). Let χ be an **LJ**-proof of a sequent S. Then there exists a proof χ' of $sk(S)$ such that $l(\chi') \leq l(\chi)$.

Proof Sketch. By replacing variables bounded by strong quantifiers by their corresponding skolem terms and skipping the corresponding quantifier introduction rules. □

Lemma 5.5 ($\exists$-Elimination). Let $\chi \in \mathcal{QMON}$ be a proof of a sequent S where all cut formulas are of $\{\wedge, \forall, \exists\}$-type and S does not contain strong quantifiers. Then there exists a proof $\chi' \in \mathcal{QMON}$ of S such that all cut formulas are of $\{\wedge, \forall\}$-type and $l(\chi') \leq l(\chi)$.

Proof. By Lemma 5.1 we can replace χ by a proof $\chi_1 \in \mathcal{QMON}^*$ of S. We iterate the following procedure for the uppermost, leftmost cut containing $\exists$ until there is no such cut anymore: By Lemma 5.2 for every critical occurrence μ of a subformula B of the form $B = (\exists x)C$ it holds that $\mathcal{P}_\mu \neq \emptyset$. After applying the cut projection method for each such μ we obtain a new cut formula without $\exists$. □

Definition 5.14 (Contraction-Normalized). A proof is called *contraction-normalized* if applications of the contraction rule only appear as last inferences (i.e. immediately above the end-sequent).

Lemma 5.6 ($\forall, \wedge$-Elimination). Let $\chi \in \mathcal{QMON}$ be a proof of a sequent $S : \Gamma \vdash F$ such that $\Gamma \vdash$ does not contain strong quantifiers and all cuts are of $\{\forall, \wedge\}$-type. Then there exists a proof $\chi' \in \mathcal{QMON}$ of S such that χ' is cut-free, contraction-normalized and $l(\chi') \leq 2^{l(\chi)}$.

Proof Sketch. By induction on the structure of χ. A cut with cut formula A is eliminated by constructing proofs from the left side that each proves an atom of A and inserting this proofs in the proof on the right side of the cut. □

Lemma 5.7 (Re-Skolemization). Let $\chi \in \mathcal{QMON}$ be a contraction normalized proof of a sequent S containing weak quantifiers only. Let S' be any sequent such that $S = sk(S')$. Then there exists a cut-free proof χ' of S' with $l(\chi') \leq (quocc(S') + 1)l(\chi)$ where *quocc* denoted the number of quantifier-occurrences.

Proof Sketch. By replacing the Skolem terms in S from the outside in by strongly quantified variables and adapting the proof above accordingly. □

Theorem 5.1 (Cut-Elimination for $\mathcal{QMON}$). Let $\chi \in \mathcal{QMON}$ be a proof of a sequent S. Then there exists a cut-free proof χ' of S such that $l(\chi') \leq size(\chi)l(\chi)2^{l(\chi)}$.

Proof. By Lemma 5.3 we can eliminate $\vee$ from the cut formulas in χ obtaining a proof $\chi_1 \in \mathcal{QMON}$ with $l(\chi_1) \leq l(\chi)$ where all cut formulas are of $\{\exists, \forall, \wedge\}$-type. By Lemma 5.4 there exists a proof $\chi_2 \in \mathcal{QMON}$ of $sk(S)$ with $l(\chi_2) \leq l(\chi_3)$, then Lemma 5.5 yields a proof $\chi_3 \in \mathcal{QMON}$ of $sk(S)$ with $l(\chi_3) \leq l(\chi_2)$ where all cut formulas are of $\{\forall, \wedge\}$-type. By Lemma 5.6 we obtain a cut-free proof $\chi_4 \in \mathcal{QMON}$ of $sk(S)$ with $l(\chi_4) \leq 2^{l(\chi_3)}$. Finally, by Lemma 5.7 we construct the proof χ' of S with

$$l(\chi') \leq l(\chi_4)(quocc(S) + 1)$$

But $quocc(S) + 1 \leq l(\chi)size(\chi)$ and so we have

$$l(\chi') \leq size(\chi)l(\chi)2^{l(\chi)}$$

□

An interesting mathematical consequence of this result is that in equational theories lemmas that do not include negative information do not have a large (i.e. non-elementary) influence on the length of proofs. This is due to the fact that we can eliminate cuts consisting of monotone (non-negative) cut formulas with only exponential expense. But if we admit negation only directly in front of atom formulas (negation normal form (NNF)) then cut-elimination is of non-elementary complexity because NNF is a normal form of PL-formulas.

5.3 Projection-Based Normalization

The cut projection method can also be applied to proofs in natural deduction although some modifications are necessary. A cut in natural deduction is a sequence of an introduction-rule and a directly succeeding elimination-rule on the same formula[3]. We will limit ourselves to →-cuts because the other types of cuts have simple reductions (∀,∧) or can easily be reduced to →-cuts (∃,∨):

$$\dfrac{\dfrac{\begin{matrix}\vdots\ \varphi(\alpha)\\ A\{x \leftarrow \alpha\}\end{matrix}}{(\forall x)A}\,\forall I}{A\{x \leftarrow t\}}\,\forall E \qquad\Longrightarrow\qquad \begin{matrix}\vdots\ \varphi(t)\\ A\{x \leftarrow t\}\end{matrix}$$

$$\dfrac{\dfrac{\begin{matrix}\vdots\ \varphi_1\\ A\end{matrix}\quad\begin{matrix}\vdots\ \varphi_2\\ B\end{matrix}}{A \wedge B}\,\wedge I}{A}\,\wedge E \qquad\Longrightarrow\qquad \begin{matrix}\vdots\ \varphi_1\\ A\end{matrix}$$

$$\dfrac{\dfrac{\begin{matrix}\vdots\ \varphi_1\\ A\{x \leftarrow t\}\end{matrix}}{(\exists x)A}\,\exists I \qquad \begin{matrix}[\![A\{x \leftarrow \alpha\}]\!]^l\\ \vdots\ \varphi_2(\alpha)\\ B\end{matrix}}{B}\,\exists E^l \qquad\Longrightarrow\qquad \dfrac{\begin{matrix}\vdots\ \varphi_1\\ A\{x \leftarrow t\}\end{matrix} \qquad \dfrac{\begin{matrix}[\![A\{x \leftarrow t\}]\!]^l\\ \vdots\ \varphi_2(t)\\ B\end{matrix}}{A\{x \leftarrow t\} \rightarrow B}\,\rightarrow I^l}{B}\,\rightarrow E$$

$$\dfrac{\dfrac{\begin{matrix}\vdots\ \varphi_1\\ A\end{matrix}}{A \vee B}\,\vee I1 \qquad \begin{matrix}[\![A]\!]^l\\ \vdots\ \varphi_2\\ C\end{matrix} \qquad \begin{matrix}[\![B]\!]^l\\ \vdots\ \varphi_3\\ C\end{matrix}}{C}\,\vee E^l \qquad\Longrightarrow\qquad \dfrac{\begin{matrix}\vdots\ \varphi_1\\ A\end{matrix} \qquad \dfrac{\begin{matrix}[\![A]\!]^l\\ \vdots\ \varphi_2\\ C\end{matrix}}{A \rightarrow C}\,\rightarrow I^l}{C}\,\rightarrow E$$

Obviously, these transformations do not increase the total length of the proof.

[3] In the typed lambda calculus, this is a redex proof term (see Definition 4.4).

Now we will define predecessor trees in **NJ**. The construction of the predecessor tree for a subformula occurrence in an **NJ**-proof is more complicated than in **LJ** because the predecessor relation is more complicated. Rules in **LJ** have only local effects: from one (or two) premise sequent(s) we can derive a conclusion sequent. In **NJ** on the other hand a rule has a global meaning in the sense that it can have effects on formulas anywhere in the proof above it (by discharging assumptions). Thus it is necessary to consider a predecessor relation that respects these "jumps" into assumptions, the usual tree predecessor relation (as in **LJ**) is not enough. It is also important to note that the predecessor of a discharged assumption lies *below* it in the proof tree. In fact, the predecessor relation in **NJ** consists of alternating upward and downward (sub-)trees in the proof tree.

Definition 5.15 (Predecessor Tree). Let φ be an **NJ**-proof and μ be a subformula occurrence in φ. We define the upwards predecessor tree $\mathcal{T}^u_\mu(\mu)$ and the downwards predecessor tree $\mathcal{T}^d_\mu(\mu)$ inductively: Let A, B, F, G be formulas with $A[F]$ and $B[G]$ and $\lambda, \gamma, \tau, \theta$ be subformula occurrences, $\circ$ means concatenation and $\langle \mathcal{T}_1, \dots, \mathcal{T}_n \rangle$ means branching:

1. non-Leaves
   We list the definitions for some rules as examples, for the other rules $\mathcal{T}^u_\mu$ and $\mathcal{T}^d_\mu$ are defined analogously.

$$\dfrac{\begin{matrix}\vdots & \vdots \\ A[F]_\lambda & B[G]_\tau\end{matrix}}{A[F]_\gamma \wedge B[G]_\theta}\wedge I \qquad \begin{matrix}\mathcal{T}^d_\mu(\lambda) = \lambda \circ \mathcal{T}^d_\mu(\gamma) & \mathcal{T}^d_\mu(\tau) = \tau \circ \mathcal{T}^d_\mu(\theta) \\ \mathcal{T}^u_\mu(\gamma) = \gamma \circ \mathcal{T}^u_\mu(\lambda) & \mathcal{T}^u_\mu(\theta) = \theta \circ \mathcal{T}^u_\mu(\tau)\end{matrix}$$

$$\dfrac{\begin{matrix}[\![A[F]_{\lambda_1}]\!]^l \dots [\![A[F]_{\lambda_n}]\!]^l \\ \vdots \\ B[G]_\tau\end{matrix}}{A[F]_\gamma \rightarrow B[G]_\theta}\rightarrow I^l \qquad \begin{matrix}\mathcal{T}^d_\mu(\tau) = \tau \circ \mathcal{T}^d_\mu(\theta) & \mathcal{T}^u_\mu(\theta) = \theta \circ \mathcal{T}^u_\mu(\tau) \\ \mathcal{T}^u_\mu(\gamma) = \gamma \circ \langle \mathcal{T}^d_\mu(\lambda_1), \dots, \mathcal{T}^d_\mu(\lambda_n) \rangle & \end{matrix}$$

$$\dfrac{\begin{matrix}\vdots & \vdots \\ A[F]_\lambda & A[F]_\gamma \rightarrow B[G]_\tau\end{matrix}}{B[G]_\theta}\rightarrow E \qquad \begin{matrix}\mathcal{T}^d_\mu(\lambda) = \lambda \circ \mathcal{T}^u_\mu(\gamma) & \mathcal{T}^d_\mu(\tau) = \tau \circ \mathcal{T}^d_\mu(\theta) \\ \mathcal{T}^d_\mu(\gamma) = \gamma \circ \mathcal{T}^u_\mu(\lambda) & \mathcal{T}^u_\mu(\theta) = \theta \circ \mathcal{T}^u_\mu(\tau)\end{matrix}$$

   etc.

2. Leaves

   (a) main-symbol closure
   Let μ be an occurrence of the subformula A. A node λ in a predecessor tree $\mathcal{T}^u_\mu$ or $\mathcal{T}^d_\mu$ is a *leaf* if it is an occurrence of the *formula* A in φ.

(b) weakening-like closure
A node in an upwards predecessor tree is a leaf if it is an occurrence in the disappearing disjunct of an $\vee I$-rule or in the conclusion formula of a $\bot E$-rule. A node in a downwards predecessor tree is a leaf if it is an occurrence in the disappearing conjunct of an $\wedge E$-rule. A node λ of an occurrence of the formula that can be discharged at a rule $\xi \in \{\vee E, \exists E, \rightarrow I, \neg I\}$ is a leaf if ξ does not discharge an assumption[4].

In natural deduction there are the same two kinds of leaves as in the sequent calculus. The above definition of the main-symbol closure leaves makes sense because if a subformula A occurs as a formula it must be decomposed in the next step[5]. A predecessor tree in **NJ** is called *closed* if every branch has a leaf.

For a node λ in an (upwards or downwards) predecessor tree $\mathcal{T}_\mu$ the *associated substitution* is defined analogously to **LJ**: $\sigma_\mu = id$ and if λ is in a upwards predecessor tree below a quantifier introduction rule or in a downwards predecessor tree above a quantifier elimination rule we add the corresponding substitution if λ is in its scope. Also the notion of critical occurrence is analogous to that in **LJ** with the only difference being that the left side of a sequent corresponds to the assumptions in **NJ** and that so there can be more than one right-occurrence of a subformula (because the $\rightarrow I$ rule of a cut can discharge more than one formula in the assumptions).

Definition 5.16 (Critical Occurrence). Let φ be an **NJ**-proof of the form

$$\dfrac{\begin{matrix} \vdots\ \varphi_1 \\ A \end{matrix} \qquad \dfrac{\begin{matrix} [\![A]\!]^l \\ \vdots\ \varphi_2 \\ F \end{matrix}}{A \rightarrow F}\ \rightarrow I^l}{F}\ \rightarrow E$$

and B be a subformula of A, μ_l be the occurrence of B in A as conclusion of φ_1 and μ_r be an occurrence of B in A as discharged assumption in φ_2.

1. B is of the form $B = C \vee D$ or $B = (\exists x)C$: Then μ_l (μ_r) is called a *critical occurrence* if it is of positive (negative) polarity in A.

2. B is of the form $B = C \wedge D$ or $B = (\forall x)C$: Then μ_l (μ_r) is called a *critical occurrence* if it is of negative (positive) polarity in A.

Definition 5.17 (Projection Targets). The set of projection targets $\mathcal{P}_\mu$ of the predecessor tree $\mathcal{T}_\mu$ of a critical subformula occurrence μ is defined as in **LJ**.

[4]Note that these cases correspond to $\vee : r$, $w : r$, $\wedge : l$ and $w : l$ in **LJ**.

[5]This is not the case in **LJ** where a formula can be carried over to another node of the proof tree without change because it is in the context of the applied rule.

Now we are ready to define cut projection for natural deduction. It works pretty much the same as in the sequent calculus with one exception: When projecting subformulas B of the form $B = C \vee D$ or $B = (\exists x)C$ then we need to introduce an additional cut on the non-critical side. This is due to the fact the the $\vee E$- and $\exists E$-rules can discharge an arbitrary number of assumptions and deleting the logical symbol (and thus the rule that eliminates it) makes it necessary to introduce another rule to discharge these assumptions.

Definition 5.18 (Cut-Projection). Let χ be an **NJ**-proof of the form

$$\dfrac{\begin{array}{c}\vdots\ \chi_1\\ A\end{array} \quad \dfrac{\begin{array}{c}[\![A]\!]^l\\ \vdots\ \chi_2\\ F\end{array}}{A \to F}\ {\to}I^l}{F}\ {\to}E$$

and B be a subformula of A, let μ be the critical and θ be the non-critical occurrence of B in A and let $\mathcal{T}_\mu$ and $\mathcal{T}_\theta$ be the closed predecessor trees of μ and θ. The projected proof χ^* is defined in the following way:

1. Select a projection target $B^* \in \mathcal{P}_\mu$
2. For all non-leaf nodes λ of $\mathcal{T}_\mu$ and $\mathcal{T}_\theta$: Replace the formula at λ by $B^*\sigma_\lambda$.
3. For all leaves λ of the critical predecessor tree $\mathcal{T}_\mu$ let ξ be the rule after λ (in upwards or downwards direction, depending on the current tree direction) and ψ be the proof ending with ξ.
   (a) main-symbol closure
   In Cases (i) and (ii) λ occurs in an upwards predecessor tree, whereas in Cases (iii) and (iv) λ occurs in a downwards predecessor tree.
      i. $\xi = \vee I1$ (or $\xi = \vee I2$)
      Then ψ is of the form

$$\dfrac{\begin{array}{c}\vdots\ \varphi\\ C'\end{array}}{C' \vee D'}\ \vee I1$$

      and B is of the form $B = C \vee D$ with $C' = C\sigma_\lambda$ and $D' = D\sigma_\lambda$ and $B^* = C$ (or $B^* = D$). We replace ψ by φ.
      ii. $\xi = \exists I$
      Then ψ is of the form

$$\dfrac{\begin{array}{c}\vdots\ \varphi\\ C'\{x \leftarrow t'\}\end{array}}{(\exists x)C'}\ \exists I$$

B is of the form $B = (\exists x)C$ with $C' = C\sigma_\lambda$ and $B^* = C\{x \leftarrow t\}$ with $t' = t\sigma_\lambda$. We replace ψ by φ.

iii. $\xi = \wedge E1$ (or $\xi = \wedge E2$)
Then ψ is of the form

$$\dfrac{\begin{array}{c}\vdots\,\varphi\\ C' \wedge D'\end{array}}{C'}\ \wedge E1$$

where φ is already projected, B is of the form $B = C \wedge D$ with $C' = C\sigma_\lambda$ and $D' = D\sigma_\lambda$ and $B^* = C$ (or $B^* = D$). We simply omit ξ to obtain a proof φ^* of C' (or D').

iv. $\xi = \forall E$
Then ψ is of the form

$$\dfrac{\begin{array}{c}\vdots\,\varphi\\ (\forall x)C'\end{array}}{C'\{x \leftarrow t'\}}\ \forall E$$

where φ is already projected, B is of the form $B = (\forall x)C$ with $C' = C\sigma_\lambda$ and $B^* = C\{x \leftarrow t\}$ with $t' = t\sigma_\lambda$. We simply skip ξ to obtain a proof φ^* of $C'\{x \leftarrow t'\}$.

(b) weakening-like closure: Replace the formula at λ by $B^*\sigma_\lambda$.

4. For all leaves λ of the non-critical predecessor tree $\mathcal{T}_\theta$. Let ξ be the rule after λ (in upwards or downwards direction, depending on the current ree direction) and ψ be the proof ending with ξ.

(a) main-symbol closure
In Cases (i) and (ii) λ occurs in a downwards predecessor tree, whereas in Cases (iii) and (iv) λ occurs in an upwards predecessor tree.

i. $\xi = \vee E$
Then ψ is of the form

$$\dfrac{\begin{array}{c}\\ \vdots\,\varphi_1\\ C' \vee D'\end{array}\quad \begin{array}{c}[\![C']\!]^l\\ \vdots\,\varphi_2\\ F\end{array}\quad \begin{array}{c}[\![D']\!]^l\\ \vdots\,\varphi_3\\ F\end{array}}{F}\ \vee E^l$$

where φ_1 is already projected, B is of the form $B = C \vee D$ with $C' = C\sigma_\lambda$ and $D' = D\sigma_\lambda$ and $B^* = C$ (or $B^* = D$). We replace ψ by

$$\dfrac{\begin{array}{c}\vdots\,\varphi_1^*\\ C'\end{array}\quad \dfrac{\begin{array}{c}[\![C']\!]^l\\ \vdots\,\varphi_2\\ F\end{array}}{C' \to F}\ \to I^l}{F}\ \to E$$

where φ_1^* is obtained from φ_1 by replacing the root $C' \vee D'$ by C' (symmetric for $B^* = D$). Note that a new cut has to be introduced here to discharge the assumption C' (or D').

ii. $\xi = \exists E$
Then ψ is of the form

$$\dfrac{\begin{matrix} \vdots\, \varphi_1 \\ (\exists x)C' \end{matrix} \qquad \begin{matrix} [\![C'\{x \leftarrow \alpha\}]\!]^l \\ \vdots\, \varphi_2(\alpha) \\ F \end{matrix}}{F}\ \exists E^l$$

where φ_1 is already projected, B is of the form $(\exists x)C$ with $C' = C\sigma_\lambda$ and $B^* = C\{x \leftarrow t\}$ for some term t. Let $t' = t\sigma_\lambda$, we replace ψ by

$$\dfrac{\begin{matrix} \vdots\, \varphi_1^* \\ C'\{x \leftarrow t'\} \end{matrix} \qquad \dfrac{\begin{matrix} [\![C'\{x \leftarrow t'\}]\!]^l \\ \vdots\, \varphi_2(t') \\ F \end{matrix}}{C'\{x \leftarrow t'\} \rightarrow F}\ \rightarrow I^l}{F}\ \rightarrow E$$

where φ_1^* is obtained from φ_1 by replacing the root $(\exists x)C'$ by $C'\{x \leftarrow t'\}$. Note that a new cut has to be introduced here to discharge the assumption $C'\{x \leftarrow t'\}$.

iii. $\xi = \wedge I$
Then ψ is of the form

$$\dfrac{\begin{matrix} \vdots\, \varphi_1 \\ C' \end{matrix} \qquad \begin{matrix} \vdots\, \varphi_2 \\ D' \end{matrix}}{C' \wedge D'}\ \wedge I$$

and B is of the form $B = C \wedge D$ with $C' = C\sigma_\lambda$, $D' = D\sigma_\lambda$ and $B^* = C$ (or $B^* = D$). We replace ψ by

$$\begin{matrix} \vdots\, \varphi_1 \\ C' \end{matrix}$$

(symmetric for $B^* = D$).

iv. $\xi = \forall I$
The ψ is of the form

$$\dfrac{\begin{matrix} \vdots\, \varphi(\alpha) \\ C'\{x \leftarrow \alpha\} \end{matrix}}{(\forall x)C'}\ \forall I$$

and B is of the form $B = (\forall x)C$ with $C' = C\sigma_\lambda$ and $B^* = C\{x \leftarrow t\}$ for some term t. Let $t' = t\sigma_\lambda$, we replace ψ by

$$\begin{array}{c} \vdots\ \varphi(t') \\ C'\{x \leftarrow t'\} \end{array}$$

(b) weakening-like closure: Replace the formula at λ by $B^*\sigma_\lambda$.

We observe the effect that cut projection in natural deduction works well only for $\wedge$- and $\forall$-subformulas of cut formulas. When dealing with $\vee$- and $\exists$-subformulas it is necessary to introduce an additional cut on the uncritical (right) side to maintain the proof. The reason for this problem is an important difference between sequent calculus and natural deduction: In **NJ** there is no explicit contraction rule, because the contractions on the left side from **LJ** correspond to the possibility of discharging more than one assumptions with one rule (and contractions on the right side are not allowed in **LJ** anyway). In the rules $\vee E$ and $\exists E$ two things happen simultaneously which are separated in the sequent calculus: 1) the deletion of a logical symbol and 2) (possibly) a contraction on the left side (by discharging more than one assumption). By transforming the proof in a way that deletes an $\vee$- or $\exists$-symbol one needs to delete the corresponding elimination rule which leads to new active assumptions on the right side of this rule. To maintain the connections between the left proof and the right proof above this rule one needs to introduce an additional cut. The only alternative is to insert the left side for every occurrence of the discharged assumption, but this is not a real alternative because it is nothing else than a βF-reduction of the proof with a new cut.

Example 5.3. This example is an illustration of the effects of cut projection on lambda terms. Consider the following **NJ**-proof:

$$\dfrac{\dfrac{\dfrac{(\forall y)P(y)}{P(\alpha)}\ \forall E}{(\forall x)P(x)}\ \forall I \qquad \dfrac{\dfrac{[\![(\forall x)P(x)]\!]^l}{P(a)}\ \forall E}{(\forall x)P(x) \rightarrow P(a)}\ \rightarrow I^l}{P(a)}\ \rightarrow E$$

The corresponding lambda term is

$$\mathsf{M} \equiv (\lambda \mathsf{x}_1.\mathsf{x}_1 a)(\lambda \mathsf{x}.\mathsf{y}\mathsf{x})$$

with

$$\mathsf{x} : i, \mathsf{x}_1 : (\forall x)P(x), \mathsf{y} : (\forall y)P(y)$$

So we have $\mathsf{M} \equiv \mathsf{M}_\mathsf{r}\mathsf{M}_\mathsf{l}$ where $\mathsf{M}_\mathsf{r} \equiv (\lambda \mathsf{x}_1.\mathsf{x}_1 a)$ and $\mathsf{M}_\mathsf{l} \equiv (\lambda \mathsf{x}.\mathsf{y}\mathsf{x})$. Note that $\mathsf{M}_\mathsf{l} : (\forall x)P(x)$ and $\mathsf{x}_1 : (\forall x)P(x)$. This proof is projected to

$$\dfrac{\dfrac{(\forall y)P(y)}{P(a)}\ \forall E \qquad \dfrac{[\![P(a)]\!]^l}{P(a) \rightarrow P(a)}\ \rightarrow I^l}{P(a)}\ \rightarrow E$$

which is

$$M' \equiv (\lambda \mathsf{x}_1.\mathsf{x}_1)(\mathsf{y}a)$$

with

$$\mathsf{x}_1 : P(a), \mathsf{y} : (\forall y)P(y)$$

Splitting M' up in the same way as M leads to $\mathsf{M}' \equiv \mathsf{M}'_\mathsf{r}\mathsf{M}'_\mathsf{l}$ where $\mathsf{M}'_\mathsf{r} \equiv (\lambda \mathsf{x}_1.\mathsf{x}_1)$ and $\mathsf{M}'_\mathsf{l} \equiv (\mathsf{y}a)$. The crucial point is that $\mathsf{M}_\mathsf{l} : P(a)$ and that also the type of x_1 has been changed so that $\mathsf{x}_1 : P(a)$.

So cut projection transforms lambda terms in a way that simplifies the (type of the) interface between two terms. This is done by imaging M_l being inserted in M_r and finding the term (in our case a) that will be applied to M_l in M_r. This application is extracted from M_r and incorporated into M_l. It is also easy to understand the uniqueness condition in this context: If we had $\mathsf{M}_\mathsf{r} = \lambda \mathsf{x}_1.\langle \mathsf{x}_1 a, \mathsf{x}_1 b\rangle$ it is clear that if we imagine M_l being inserted for x_1 there is no unique term that will be applied to M_l in M_r. On the other hand, there would exists such a term if we had $\mathsf{M}_\mathsf{r} = \lambda \mathsf{x}_1.\langle \mathsf{x}_1 a, \mathsf{x}_1 a\rangle$.

Example 5.4. It is - in principle - also possible to use this idea of interface simplification for “real”, i.e. industrial programming languages, like for example `C`: Consider the following `C`-Code:

```
struct person {
    char* name ;
    char* address ;
} ;

void print_address (struct person p)
{
    printf("address: %s\n", p.address) ;
}

void print_name (struct person p)
{
    printf("name: %s\n", p.name) ;
}

void print_person (struct person p)
{
    print_name(p) ;
    print_address(p) ;
}
```

What can be observed here is that the procedures `print_name` and `print_address` have too general interfaces because they take a `person`-structure but only need a string: the name or the address. A `struct` in `C` corresponds to a conjunction (as means of constructing a tuple type). We can view `print_name` and `print_address` as two right sides of two cuts located in the `print_person` procedure. The fact that we need only *one* of the `name`- and `address`-fields in each procedure corresponds to the `person`-occurrences in these cuts fulfilling the uniqueness condition, because when we are dealing with a conjunction the critical side is the right side (i.e. the called procedure). By simplifying these interface types in a way inspired by cut projection we would obtain:

```
struct person {
    char* name ;
    char* address ;
} ;

void print_address (char* p)
{
    printf("address: %s\n", p) ;
}

void print_name (char* p)
{
    printf("name: %s\n", p) ;
}

void print_person (struct person p)
{
    print_name(p.name) ;
    print_address(p.address) ;
}
```

Bibliography

[1] Matthias Baaz and Alexander Leitsch. Fast Cut-Elimination by Projection. In Dirk van Dalen and Marc Bezem, editors, *Proceedings Computer Science Logic '96*, volume 1258 of *Lecture Notes in Computer Science*, pages 18–33. Springer, 1996.

[2] Matthias Baaz and Alexander Leitsch. Cut Normal Forms and Proof Complexity. *Annals of Pure and Applied Logic*, 97:127–177, 1999.

[3] Matthias Baaz and Alexander Leitsch. Cut-elimination and Redundancy-elimination by Resolution. *Journal of Symbolic Computation*, 29(2):149–176, 2000.

[4] Matthias Baaz and Alexander Leitsch. Comparing the Complexity of Cut-Elimination Methods. In R. Kahle, P. Schroeder-Heister, and R. Stärk, editors, *Proof Theory in Computer Science '01*, volume 2183 of *Lecture Notes in Computer Science*, pages 49–67. Springer, 2001.

[5] Hendrik Pieter Barendregt. *The Lambda Calculus*, volume 103 of *Studies in Logic and the Foundations of Mathematics.* Elsevier, 1984.

[6] Hendrik Pieter Barendregt. *Lambda Calculi with Types*, volume 2 of *Handbook of Logic in Computer Science*, pages 117–309. Clarendon Press, Oxford, 1992.

[7] G. Gentzen. Untersuchungen über das logische Schließen. *Mathematische Zeitschrift*, 39:176–210,405–431, 1934–1935.

[8] Jean-Yves Girard. *Proof Theory and Logical Complexity.* Elsevier, 1987.

[9] Jean-Yves Girard, Paul Taylor, and Yves Lafont. *Proofs and Types.* Cambridge University Press, 1989.

[10] P.V. Orevkov. Lower bounds for increasing complexity of derivations after cut elimination. *Zapiski Nauchnykh Seminarov Leningradskogo Otdeleniya Matematicheskogo Instituta*, 88:137–161, 1979.

[11] R. Statman. Lower bounds on Herbrand's theorem. *Proceedings of the American Mathematical Society*, 75:104–107, 1979.

[12] Sören Stenlund. *Combinators, Lambda Terms and Proof Theory.* Synthese Library. D. Reidel/Dodrecht, 1972.

[13] W.W. Tait. Normal derivability in classical logic. In Jon Barwise, editor, *The Syntax and Semantics of Infinitary Languages*, pages 204–236. Springer, Berlin, 1968.

[14] Gaisi Takeuti. *Proof Theory.* North-Holland, Amsterdam, 2nd edition, March 1987.

[15] J. Zucker. The Correspondence Between Cut-Elimination and Normalization. *Annals of Mathematical Logic*, 7:1–112, 1974.