....
Alexander Leitsch

TECHNISCHE
UNIVERSITÄT
WIEN

VIENNA
UNIVERSITY OF
TECHNOLOGY

M.Sc. Arbeit

Herbrand Sequent Extraction

ausgeführt am

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

unter der Anleitung von
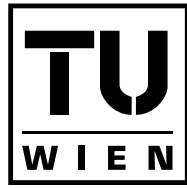
Univ.Prof. Dr.phil. Alexander Leitsch

durch

Bruno Woltzenlogel Paleo

Wien, 04. Mai 2007

....
Bruno Woltzenlogel Paleo

. . . .

Alexander Leitsch

TECHNISCHE
UNIVERSITÄT
WIEN

VIENNA
UNIVERSITY OF
TECHNOLOGY

MASTER THESIS

Herbrand Sequent Extraction

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

Bruno Woltzenlogel Paleo

Vienna, May 04, 2007

. . . .

Bruno Woltzenlogel Paleo

Contents

Kurzfassung

Nach der Definition einer Verallgemeinerung des Herbrandschen Theorems für Sequente, beschreiben wir drei schon existierende Algorithmen für die Extraktion eines Herbrandsequents aus formalen Beweisen, die im Sequentialkalkül **LK** geschrieben sind. Darüber hinaus verbessern wir diese drei Algorithmen und entwicklen daraus einen vierten Algorithmus, welcher deren Ideen vereinigt und generalisiert. Dieser Algorithmus wurde dann realisiert und in das CERes (Cut-Elimination by Resolution) Projekt integriert. Die Wichtigkeit der Extraktion von Herbrandsequenten liegt darin, dass Herbrandsequente die Kreativität der Beweise enthält.

Abstract

After defining a generalization of Herbrand's Theorem for sequents, we describe three pre-existing algorithms for the extraction of a Herbrand sequent of the end-sequent of proofs in the Sequent Calculus **LK**. Furthermore, we improve these three algorithms by designing a new fourth algorithm, which combines and generalizes their essential ideas. The implementation of this new algorithm was realized within the framework of the project CERes (Cut-Elimination by Resolution). The importance of extracting Herbrand sequents from proofs lies on the fact that a Herbrand sequent summarizes the creative content of a proof.

Dedication

I dedicate this master thesis to my parents, who in spite of the distance, continued to support me with their love and attention.

Acknowledgements

Chapter 1

Introduction

In Proof Theory, we are interested in constructing and analyzing *formal proofs*, which strictly follow *axioms* and *rules of inference* of formal logical proof calculi, as Hilbert Calculi, Natural Deduction Calculi or Sequent Calculi. Some of the main advantages of the formalization of proofs are:

- The correctness of formal proofs can be easily checked, by verifying whether the formal axioms and rules of the calculus were correctly employed.

- Formal proofs for formalized statements (formulas) may be constructed by computers executing Automatic Theorem Provers, as in [11].

- New informal proofs for statements may be discovered by applying formal transformations to their corresponding formal proofs, as in [3] and [4].

- Formal proofs, when viewed and studied as a model and ideal for informal mathematical proofs, allow meta-mathematical investigations into the foundations of Mathematics.

However, formal representations of real informal mathematical proofs or computer generated proofs of real mathematical problems usually have some drawbacks that make them difficult to be analyzed and understood by mathematicians or by humans in general. Firstly, the size of a formal proof is usually huge ([2]), which makes it hard to be visualized in its whole. Secondly, many of its individual inferences are only structural, necessary not to carry some essential idea about the proof, but only to satisfy the formalities of the calculus used. Thirdly, inference rules of proof calculi not always correspond easily to natural inferences in informal proofs. Together these drawbacks imply that, given a formal proof, it is not easy for a human to extract its essential idea, because one gets lost in a proof which looks like a huge data structure of repetitive, bureaucratic and non-intuitive formalities

that involuntarily hide the crucial information. Therefore there is a clear
need for summarization of formal proofs or for extraction of its essential
ideas, whenever these proofs are intended to be analyzed and understood by
humans.

This thesis describes one possible technique that helps to fulfill this need
in the particular case of First-Order Logic and Sequent Calculus **LK**. Our
technique relies on the concept of Herbrand Sequent (a generalization of
Herbrand Disjunction ([9]) and its extraction from **LK**-Proofs in a way that
summarizes their creative content, which lies on the instantiations chosen
for quantified variables.

The following chapters form this thesis:

- Chapter 2, *Notations and Definitions*, defines the language used for
  First-Order Logic and the variant of Sequent Calculus **LK** used.

- Chapter 3, *Proof Analysis and Transformations*, defines the concept
  of Herbrand sequent and describes important and related results, as
  Herbrand's Theorem and Gentzen's Mid-Sequent Theorem, which are
  essential to the methods described in the next chapter.

- Chapter 4, *Extraction of Herbrand Sequents*, has four subsections, each
  devoted to a different algorithm for the extraction of Herbrand Se-
  quents.

The novel contribution made by this thesis is the fourth algorithm, *Ex-
traction via Collection of Sub-Formula Instances*, which combines and gen-
eralizes the three pre-existing algorithms.

Chapter 2

Notations and Definitions

The notations and definitions described in this chapter construct the basic formal language and the formal proof calculus which are used in this thesis: First-Order Logic and the Sequent Calculus **LK**. Together they allow the formalization of proofs.

2.1 Meta-Notations

The symbol (=) is usually used ambiguously in science. Here we reduce this ambiguity by using the two different symbols below:

- $\doteq$: is used to define a simpler notation for a, usually more complex, existing expression. Thus $A \doteq B$ implies that we may use A as a shorthand notation for B.

- =: is used to assert that two expressions are syntactically equal.

2.2 First-Order Logic

In this section we firstly describe a language for First-Order Logic, which consists of an alphabet and formation rules to construct well-formed terms and formulas from these.

Secondly, notations for the manipulation of formulas are described. These notations allow reference to sub-formulas in specific positions of the formula and substitution of variables or sub-formulas in formulas.

And finally, further definitions are given, which distinguish and classify special properties owned by certain sub-formulas, quantifiers and variables. These classifications are important for the definition of the algorithms that will be described in the subsequent chapters.

3

2.2.1   The Language

Definition 2.2.1 (Alphabet of First-Order Logic)**.** The *Alphabet* of First-Order Logic consists of:

1. A countable set of variables V

2. A countable set of constant symbols C

3. For every $n \geq 1$, a countable set of function symbols F_n. ($F \doteq \bigcup_{n \geq 1} F_n$ is the set of all function symbols).

4. For every $n \geq 1$, a countable set of predicate symbols P_n. ($P \doteq \bigcup_{n \geq 1} P_n$ is the set of all predicate symbols).

5. A set of propositional connectives $\{\vee, \wedge, \rightarrow, \neg\}$

6. A set of quantifiers $\{\forall, \exists\}$

7. The set of parentheses $\{(,)\}$

8. The set of symbols $\{\top, \bot\}$

Definition 2.2.2 (Terms)**.** The set of *terms* T is defined as the smallest set satisfying:

1. $V \subset T$

2. $C \subset T$

3. For all $n \geq 1$: If $f \in F_n$ and $t_1, \ldots, t_n \in T$, then $f(t_1, \ldots, t_n) \in T$.

Definition 2.2.3 (Formulas)**.** The set of *formulas* F_{FOL} is defined as the smallest set satisfying:

1. For all $n \geq 1$: If $P \in P_n$ and $t_1, \ldots, t_n \in T$, then $P(t_1, \ldots, t_n) \in F_{FOL}$.

2. If $A \in F_{FOL}$, then $\neg A \in F_{FOL}$

3. If $A, B \in F_{FOL}$, then $(A \wedge B) \in F_{FOL}$

4. If $A, B \in F_{FOL}$, then $(A \vee B) \in F_{FOL}$

5. If $A, B \in F_{FOL}$, then $(A \rightarrow B) \in F_{FOL}$

6. If $A \in F_{FOL}$ and $x \in V$, then $(\forall x)A \in F_{FOL}$

7. If $A \in F_{FOL}$ and $x \in V$, then $(\exists x)A \in F_{FOL}$

For the sake of transparency, we usually omit the outermost parentheses of a formula.

2.2.2 Manipulation of Formulas

Definition 2.2.4 (Positions of a Formula). Let A be a formula, then its set of *positions* $pos(A)$ is the smallest set of strings belonging to $\{1,2\}^*$ and such that $pos(A)$ satisfies:

1. $\varepsilon \in pos(A)$

2. If A is of the form $\neg B$, then $\{1.\lambda | \lambda \in pos(B)\} \subset pos(A)$.

3. If A is of the form $(B \wedge C)$, then $(\{1.\lambda_1 | \lambda_1 \in pos(B)\} \cup \{2.\lambda_2 | \lambda_2 \in pos(C)\}) \subset pos(A)$.

4. If A is of the form $(B \vee C)$, then $(\{1.\lambda_1 | \lambda_1 \in pos(B)\} \cup \{2.\lambda_2 | \lambda_2 \in pos(C)\}) \subset pos(A)$.

5. If A is of the form $(B \rightarrow C)$, then $(\{1.\lambda_1 | \lambda_1 \in pos(B)\} \cup \{2.\lambda_2 | \lambda_2 \in pos(C)\}) \subset pos(A)$.

6. If A is of the form $(\forall x)B$, then $\{1.\lambda | \lambda \in pos(B)\} \subset pos(A)$.

7. If A is of the form $(\exists x)B$, then $\{1.\lambda | \lambda \in pos(B)\} \subset pos(A)$.

Example 2.1 (Positions of a Formula). *Below are the sets of positions for some example formulas:*

1. *$pos(F(a) \vee G(b)) = \{\varepsilon, 1, 2\}$*

2. *$pos(F(a) \vee (G(b) \wedge F(b))) = \{\varepsilon, 1, 2, 21, 22\}$*

3. *$pos((\forall x)F(x) \vee (G(b) \wedge F(b))) = \{\varepsilon, 1, 2, 11, 21, 22\}$*

4. *$pos(F(c) \vee (\exists x)(G(x) \wedge F(b))) = \{\varepsilon, 1, 2, 21, 211, 212\}$*

Definition 2.2.5 (Depth of a Position of a Formula). Let λ be a position of a formula F. Then, the *depth* of λ, $depth(\lambda)$ is defined inductively:

1. $depth(\epsilon) \doteq 0$.

2. $depth(1.\lambda') \doteq 1 + depth(\lambda')$.

3. $depth(2.\lambda') \doteq 1 + depth(\lambda')$.

Definition 2.2.6 (Atomic Formulas, Compound Formulas). A formula A is *atomic* if $pos(A) = \{\varepsilon\}$. In this case, A is clearly of the form $P(t_1, \ldots, t_n)$, where $P \in P_n$ and $t_1, \ldots, t_n \in T$. Otherwise, it is said to be *compound*.

Definition 2.2.7 (Sub-formulas occurring in given Positions). Let λ be a position of the formula A. Then the *sub-formula* occurring in position λ is denoted $|A|_\lambda$. Formally, we may define the computation of $|A|_\lambda$ inductively:

1. $|A|_\varepsilon \doteq A$

2. If A is of the form $\neg B$ and $\lambda = 1.\lambda'$ with $\lambda' \in pos(B)$, then $|A|_\lambda \doteq |B|_{\lambda'}$.

3. If A is of the form $B \circ C$ (for $\circ \in \{\wedge, \vee, \rightarrow\}$) and $\lambda = 1.\lambda'$ with $\lambda' \in pos(B)$, then $|A|_\lambda \doteq |B|_{\lambda'}$.

4. If A is of the form $B \circ C$ (for $\circ \in \{\wedge, \vee, \rightarrow\}$) and $\lambda = 2.\lambda'$ with $\lambda' \in pos(C)$, then $|A|_\lambda \doteq |C|_{\lambda'}$.

5. If A is of the form $(Qx)B$ (for $Q \in \{\forall, \exists\}$) and $\lambda = 1.\lambda'$ with $\lambda' \in pos(B)$, then $|A|_\lambda \doteq |B|_{\lambda'}$.

Example 2.2 (Sub-formulas occurring in given Positions)**.** *Below are some examples of sub-formulas occurring in specified positions of the formula $F(c) \vee (\exists x)(G(x) \wedge F(b))$:*

   1. $|F(c) \vee (\exists x)(G(x) \wedge F(b))|_\varepsilon = F(c) \vee (\exists x)(G(x) \wedge F(b))$

   2. $|F(c) \vee (\exists x)(G(x) \wedge F(b))|_1 = F(c)$

   3. $|F(c) \vee (\exists x)(G(x) \wedge F(b))|_2 = (\exists x)(G(x) \wedge F(b))$

   4. $|F(c) \vee (\exists x)(G(x) \wedge F(b))|_{21} = (G(x) \wedge F(b))$

   5. $|F(c) \vee (\exists x)(G(x) \wedge F(b))|_{211} = G(x)$

   6. $|F(c) \vee (\exists x)(G(x) \wedge F(b))|_{212} = F(b)$

Definition 2.2.8 (Sub-formula Substitution)**.** Let λ be a position of the formula A such that $B \doteq |A|_\lambda$. The formula obtained by substituting the occurrence of B in position λ in A by C is denoted $A_\lambda[C]$ or $A\sigma^{fo}$, where $\sigma^{fo} \doteq \{\lambda/C\}$

Example 2.3 (Sub-formula Substitutions in a Formula)**.** *Below are some examples of substitutions of sub-formulas occurring in specified positions of the formula $F(c) \vee (\exists x)(G(x) \wedge F(b))$ by the formula $S(s)$:*

   1. $(F(c) \vee (\exists x)(G(x) \wedge F(b)))_\varepsilon[S(s)] = S(s)$

   2. $(F(c) \vee (\exists x)(G(x) \wedge F(b)))_1[S(s)] = S(s) \vee (\exists x)(G(x) \wedge F(b))$

   3. $(F(c) \vee (\exists x)(G(x) \wedge F(b)))_2[S(s)] = F(c) \vee S(s)$

   4. $(F(c) \vee (\exists x)(G(x) \wedge F(b)))_{21}[S(s)] = F(c) \vee (\exists x)S(s)$

   5. $(F(c) \vee (\exists x)(G(x) \wedge F(b)))_{211}[S(s)] = F(c) \vee (\exists x)(S(s) \wedge F(b))$

   6. $(F(c) \vee (\exists x)(G(x) \wedge F(b)))_{212}[S(s)] = F(c) \vee (\exists x)(G(x) \wedge S(s))$

2.2.3 Classifications

Definition 2.2.9 (Positive and Negative Sub-formulas)**.** Let B be a sub-formula of a First-Order Logic formula A. We classify B as a *positive* or *negative* Sub-formula of A according to the following cases:

- If $B = A$, then B is a positive sub-formula of A.

- If $A = \neg A_1$ and B is a positive sub-formula of A_1, then B is a negative sub-formula of A.

- If $A = \neg A_1$ and B is a negative sub-formula of A_1, then B is a positive sub-formula of A.

- If $A = (A_1 \vee A_2)$ and B is a positive sub-formula of A_1, then B is a positive sub-formula of A.

- If $A = (A_1 \vee A_2)$ and B is a positive sub-formula of A_2, then B is a positive sub-formula of A.

- If $A = (A_1 \vee A_2)$ and B is a negative sub-formula of A_1, then B is a negative sub-formula of A.

- If $A = (A_1 \vee A_2)$ and B is a negative sub-formula of A_2, then B is a negative sub-formula of A.

- If $A = (A_1 \wedge A_2)$ and B is a positive sub-formula of A_1, then B is a positive sub-formula of A.

- If $A = (A_1 \wedge A_2)$ and B is a positive sub-formula of A_2, then B is a positive sub-formula of A.

- If $A = (A_1 \wedge A_2)$ and B is a negative sub-formula of A_1, then B is a negative sub-formula of A.

- If $A = (A_1 \wedge A_2)$ and B is a negative sub-formula of A_2, then B is a negative sub-formula of A.

- If $A = (A_1 \rightarrow A_2)$ and B is a positive sub-formula of A_1, then B is a negative sub-formula of A.

- If $A = (A_1 \rightarrow A_2)$ and B is a positive sub-formula of A_2, then B is a positive sub-formula of A.

- If $A = (A_1 \rightarrow A_2)$ and B is a negative sub-formula of A_1, then B is a positive sub-formula of A.

- If $A = (A_1 \rightarrow A_2)$ and B is a negative sub-formula of A_2, then B is a negative sub-formula of A.

Definition 2.2.10 (Strong and Weak Quantifiers)**.** Let $B \doteq (Qx)B'$ be a sub-formula of A. We classify Q as *strong* or *weak* according to the following cases:

- If $Q = \forall$ and B is a positive sub-formula of A, then Q is strong in A.

- If $Q = \forall$ and B is a negative sub-formula of A, then Q is weak in A.

- If $Q = \exists$ and B is a positive sub-formula of A, then Q is weak in A.

- If $Q = \exists$ and B is a negative sub-formula of A, then Q is strong in A.

Definition 2.2.11 (Scope of Quantifiers, Bound and Free Variables)**.** Let A be a formula of the form $(Qx)B$, with $Q \in \{\forall, \exists\}$ and $B \in F_{FOL}$. B and all its sub-formulas and variables are said to be in the *scope* of (Qx). A variable x is said to be *bound* if it is in the scope of (Qx). A variable x is said to be *free* if it is not in the scope of any (Qx).

Definition 2.2.12 (Variable Substitution)**.** $\sigma \doteq \{v_1 \leftarrow t_1, \ldots, v_n \leftarrow t_n\}$ denotes a variable substitution. $A\sigma$ is the formula obtained from A by substituting all occurrences of free-variables v_j by the corresponding terms t_j.

Definition 2.2.13 (Unifier)**.** Let A and B be formulas. If there is a variable substitution σ such that $A\sigma = B\sigma$, then A and B are said to be *unifiable* and σ is called a *unifier* for A and B.

2.3   Sequent Calculus

A Sequent Calculus is one of the possible calculi one may use to define the concept of *provability* of First-Order Logic formulas. The first two variants of sequent calculi were invented by Gentzen in 1934 [6], and they were called **LK**, for classical First-Order Logic, and LJ, for intuitionistic First-Order Logic.

Here we use a variant of Gentzen's Sequent Calculus **LK**. The main difference of our variant is the allowance of arbitrary atomic sequents as axiom sequents. This is an important feature, because real mathematical theories usually have a huge number of trivial (from the logical point of view, atomic) axioms, and it would be impractical to carry all these axioms in the antecedent of sequents in our **LK**-Proofs. [1]. Allowing arbitrary axiom

---

[1]For example, in order to prove a formula F within a theory that assumes the atoms A and B among other formulas $(F1, \ldots, F_n)$, in traditional Sequent Calculus **LK** we would try to find an **LK**-Proof for the sequent $A, B, F1, \ldots, F_n \vdash F$, and this **LK**-Proof would repetitively carry the atoms A and B along its branches. With our variant of the calculus **LK**, we may find an **LK**-Proof for the sequent $F_1, \ldots, F_n \vdash F$ which possibly admits the sequents $\vdash A$ and $\vdash B$ as axiom sequents.

sequents in the calculus makes the **LK**-Proofs shorter and more readable, since they do not display so much trivial logical information.

However, arbitrary axiom sequents also bring two disadvantages. Firstly, derived end-sequents do not directly correspond to valid First-Order Logic formulas anymore, but to First-Order Logic formulas that are valid relatively to the allowed axioms. Secondly, Gentzen's famous cut-elimination theorem holds only in a weaker form. Since his elimination rule for a cut on an axiom requires axiom sequents of the form $A \vdash A$, atomic cuts cannot be eliminated in our variant of the Sequent Calculus **LK**, which allows arbitrary atomic axiom sequents.

Furthermore we provide definitional and equality rules, which bring the Sequent Calculus closer to real mathematical proofs.

We define our variant of the Sequent Calculus **LK** incrementally. In the first subsection we give definitions that construct sequents from First-Order Logic formulas. In the second subsection, rules of inference are defined and constructed with sequents, and the admissible schemata of rules of inference in our **LK** is presented. In the third subsection, **LK**-Proofs are defined using rules of inference as their building blocks. Finally, a fourth chapter describes notations used to refer to specific parts of an **LK**-Proof or to denote substitutions of these parts by others.

2.3.1 Sequents

Definition 2.3.1 (Sequent)**.** A *sequent* is a pair of tuples of formulas. The first tuple is the *antecedent* of the sequent and the second tuple is the *consequent* of the sequent. We use the notation $A_1, \ldots, A_n \vdash B_1, \ldots, B_m$, where $A_1, \ldots, A_n$ is the antecedent and $B_1, \ldots, B_m$ is the consequent. The antecedent and the consequent of a sequent may also be empty ($n \geq 0$ and $m \geq 0$). If all the formulas in the antecedent and in the consequent are atomic formulas or negated atomic formulas, then the sequent is called *atomic*.

We use the symbols Γ, Π and Δ, possibly with subscripts, to denote sequences of formulas in the antecedent and consequent of sequents.

Definition 2.3.2 (Sequent Formation)**.** Given two tuples A and C of formulas, we denote by $seq(A, C)$ the sequent having A as its antecedent and C as its consequent. Given two sets or multi-sets $S_A \doteq \{A_1, \ldots, A_n\}$ and $S_C \doteq \{C_1, \ldots, C_m\}$ of formulas, we denote by $seq(S_A, S_C)$ a permutation variant of the sequent $A_1, \ldots, A_n \vdash C_1, \ldots, C_m$.

Definition 2.3.3 (Sequent Composition)**.** Let $s \doteq A_1, \ldots, A_n \vdash B_1, \ldots, B_m$ and $s' \doteq A'_1, \ldots, A'_{n'} \vdash B'_1, \ldots, B'_{m'}$. The *composition* of s and s', $A_1, \ldots, A_n, A'_1, \ldots, A'_{n'} \vdash B_1, \ldots, B_m, B'_1, \ldots, B'_{m'}$ is denoted $s \circ s'$.

Definition 2.3.4 (Subsequent)**.** Let s_1 and s_2 be sequents. s_1 is a *subsequent* of s_2, if and only there is s' such that $s_1 \circ s'$ is a permutation variant of s_2.

Definition 2.3.5 (Positions and Formula Occurrences in a Sequent)**.** Let s be a sequent with n formulas in the antecedent and m formulas in the consequent. A position in a sequent is a string of the form $c.k$ where $c \in \{A, C\}$ and k is a number such that $k \in \{0, 1, \dots, n\}$, if $c = A$, or $k \in \{0, 1, \dots, m\}$, if $c = B$. c indicates whether the position refers to a formula in the antecedent or in the consequent of the sequent, and k indicates the index of the referred formula in the antecedent or in the consequent. By $|s|_\mu$ we denote the formula occurring in position μ of s. By convention, $|s|_{A.0} \doteq \top$ and $|s|_{C.0} \doteq \bot$.

Example 2.4 (Positions and Formula Occurrences in a Sequent)**.** *Let $s \doteq A, B, C \vdash D, E$. Then:*

- $|s|_{A.2} = B$

- $|s|_{A.3} = C$

- $|s|_{C.1} = D$

Definition 2.3.6 (Occurrence Substitution in a Sequent)**.** By $s_\mu[B]$ we denote the sequent obtained from s by substituting the formula occurrence in position μ by B. Alternatively, $s_\mu[B]$ may also be denoted $s\sigma^{os}$ with $\sigma^{os} \doteq \{\mu/B\}$

Example 2.5 (Occurrence Substitution in a Sequent)**.** *Let $s \doteq A, B, C \vdash D, E$. Then:*

- $s_{A.2}[F] = s\{A.2/F\} = A, F, C \vdash D, E$

- $s_{A.3}[F] = s\{A.3/F\} = A, B, F \vdash D, E$

- $s_{C.1}[F] = s\{C.1/F\} = A, B, C \vdash F, E$

Definition 2.3.7 (Sub-formula Position in a Sequent)**.** Previously defined notations allow us to denote a sub-formula in position λ of a formula occurrence in position μ of a sequent s by the expression $||s|_\mu|_\lambda$. Alternatively we define the following simpler notation to express this same sub-formula: $|s|_{(\mu, \lambda)}$.

Definition 2.3.8 (Sub-formula Substitution in an Occurrence in a Sequent)**.** Previously defined notations allow us to denote the result of substituting a sub-formula in position λ of a formula occurrence in position μ of a sequent s by the formula A by the expression $s_\mu[(|s|_\mu)_\lambda[A]]$. Alternatively we define the following two simpler notations to express this same result:

- $s_{(\mu,\lambda)}[A]$.

- $s\sigma^{fs}$, where $\sigma^{fs} \doteq \{(\mu,\lambda)/A\}$

Definition 2.3.9 (Positive and Negative Occurrences in a Sequent)**.** Let $s = A_1, \ldots, A_n \vdash B_1, \ldots, B_m$ be a sequent. We say that $A_1, \ldots, A_n$ are *negative* occurrences in s and $B_1, \ldots, B_m$ are *positive* occurrences in s.

Definition 2.3.10 (Positive and Negative Sub-formulas in a Sequent)**.** Let s be a sequent, A be an occurrence of s and B be a sub-formula of A.

- If A occurs positively in s and B is a positive sub-formula of A, then B is a positive sub-formula of s.

- If A occurs negatively in s and B is a positive sub-formula of A, then B is a negative sub-formula of s.

- If A occurs positively in s and B is a negative sub-formula of A, then B is a negative sub-formula of s.

- If A occurs negatively in s and B is a negative sub-formula of A, then B is a positive sub-formula of s.

Definition 2.3.11 (Strong and Weak Quantifiers in a Sequent)**.** Let s be a sequent, A be an occurrence of s and Q be quantifier in A.

- If A occurs positively in s and Q is a strong quantifier in A, then Q is a strong quantifier in s.

- If A occurs positively in s and Q is a weak quantifier in A, then Q is a weak quantifier in s.

- If A occurs negatively in s and Q is a strong quantifier in A, then Q is a weak quantifier in s.

- If A occurs negatively in s and Q is a weak quantifier in A, then Q is a strong quantifier in s.

2.3.2 Inference Rules

Definition 2.3.12 (Rule)**.** A *rule* of Sequent Calculus can be of any of the three forms below:

$$\frac{}{s_C}\ r$$

$$\frac{s_P}{s_C}\ r$$

$$\frac{s_{P_1} \quad s_{P_2}}{s_C}\ r$$

In these rule schemes, we say that s_C is the *conclusion sequent*. s_P, s_{P_1}, s_{P_2} are *premise sequents*. Rules are classified according to the number of premise

sequents. Rules of the first form are *nullary rules*. Rules of the second form are *unary rules*. And rules of the third form are *binary rules*

The following rules are used in our version of **LK**:

1. axiom

$$\overline{s_A} \; axiom$$

where s_A is an arbitrary atomic sequent. We call the conclusion sequent of an axiom rule *axiom*. In particular, axioms of the form $A \vdash A$ for A atomic correspond to logical tautologies [2].

2. propositional

$$\frac{\Gamma \vdash \Delta, A \quad \Pi \vdash \Lambda, B}{\Gamma, \Pi \vdash \Delta, \Lambda, A \wedge B} \; \wedge : r$$

$$\frac{A, B, \Gamma \vdash \Delta}{A \wedge B, \Gamma \vdash \Delta} \; \wedge : l$$

$$\frac{A, \Gamma \vdash \Delta \quad B, \Pi \vdash \Lambda}{A \vee B, \Gamma, \Pi \vdash \Delta\Lambda} \; \vee : l$$

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

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

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

3. first-order

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

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

For the $\forall : r$ and the $\exists : l$ rules the variable α must not occur in Γ nor in Δ nor in A. This is the *eigenvariable condition*

For the $\forall : l$ and the $\exists : r$ rules the term t must not contain a variable that is bound in A.

---

[2]in Gentzen's original sequent calculi, only axioms of the form $F \vdash F$, where F is any arbitrary formula, were allowed.

4. structural

$$\frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, A_1, \ldots, A_n} \; w:r \qquad \frac{\Gamma \vdash \Delta}{A_1, \ldots, A_n, \Gamma \vdash \Delta} \; w:l$$

where $n > 0$

$$\frac{\Gamma \vdash \Delta, A, \Pi}{\Gamma \vdash \Delta, A, \ldots, A, \Pi} \; c:r$$

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

$$\frac{A_1, \ldots, A_n \vdash \Delta}{A_{\sigma(1)}, \ldots, A_{\sigma(n)} \vdash \Delta} \; \pi(\sigma):l \qquad \frac{\Gamma \vdash A_1, \ldots, A_n}{\Gamma \vdash A_{\sigma(1)}, \ldots, A_{\sigma(n)}} \; \pi(\sigma):r$$

where σ is a permutation given as list of cycles

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

2.3.3 LK-Proofs

Definition 2.3.13 (**LK**-Proof)**.** an **LK**-Proof is a tree in which:

1. Each node is a rule.

2. Leaves are axiom rules.

3. For every node r, each of its premise sequents is the conclusion sequent of a direct ancestor [3] node. of r.

Definition 2.3.14 (End-sequent of a proof)**.** The conclusion sequent of the root of a proof φ is called *end-sequent* and is denoted $endseq(\varphi)$.

Definition 2.3.15 (Valid Sequent)**.** A sequent s is said to be a *valid sequent*, if and only if there is an **LK**-Proof φ such that $endseq(\varphi) = s$.

Definition 2.3.16 (Alternative view of an **LK**-Proof)**.** Alternatively and more informally, an **LK**-Proof can also be viewed as a tree where the nodes are sequents, the leaves are axioms, and the edges are rules connecting the conclusions with the premises.

---

[3]The tree is drawn in the standard way, with the root below and the leaves above. When we say that a node m is the ancestor of a node n, we mean that n is in the path between m an the root.

Definition 2.3.17 (Regular Proofs)**.** An **LK**-Proof is called *regular* if and only if for each pair of strong quantifier rules ρ_1 and ρ_2, the eigen-variable of ρ_1 is different from the eigen-variable of ρ_2.

Definition 2.3.18 (Logical Equivalence of Formulas)**.** Two Formulas A and B are said to be equivalent, $A \sim B$, if and only if there is an **LK**-Proof φ such that the end-sequent of φ is $\vdash (A \rightarrow B) \wedge (B \rightarrow A)$.

Definition 2.3.19 (Logical Equivalence of Sequents)**.** Let $s \doteq A_1, \ldots, A_n \vdash B_1, \ldots, B_m$ and $s' \doteq A'_1, \ldots, A'_j \vdash B'_1, \ldots, B'_k$ be sequents. Then s and s' are said to be equivalent, if and only if there is an **LK**-Proof φ such that $endseq(\varphi) = \vdash (F_1 \rightarrow F_2) \wedge (F_2 \rightarrow F_1)$, where $F_1 \doteq (\ldots (A_1 \wedge \ldots) \wedge A_n) \rightarrow (\ldots (B_1 \vee \ldots) \vee B_m)$ and $F_2 \doteq (\ldots (A'_1 \wedge \ldots) \wedge A'_j) \rightarrow (\ldots (B'_1 \vee \ldots) \vee B'_k)$.

Definition 2.3.20 (Logical Equivalence between Formulas and Sequences of Formulas)**.** Let $A_1, \ldots, A_n$ be a sequence of formulas (occurrences or sub-formulas) in a sequent s, and let A be a formula. Then:

- $A_1^+, \ldots, A_n^+ \sim A$, if and only if $A_1, \ldots, A_n$ occur positively in s and $A \sim (A_1 \vee (\ldots (A_{n-1} \vee A_n)))$

- $A_1^-, \ldots, A_n^- \sim A$, if and only if $A_1, \ldots, A_n$ occur negatively in s and $A \sim (A_1 \wedge (\ldots (A_{n-1} \wedge A_n)))$

2.3.4   Positions and Substitutions in Proofs

Definition 2.3.21 (Positions of an **LK**-Proof)**.** Let φ be an **LK**-Proof. Then its set of *positions* $LKpos(\varphi]$ is the smallest set of strings belonging to $\{1, 2\}^*$ and such that $LKpos(\varphi)$ satisfies:

- $\varepsilon \in LKpos(\varphi)$

- If the root of φ is a unary rule, then $\{1.\nu | \nu \in LKpos(\varphi')\} \subset LKpos(\varphi)$ where φ' is the sub-tree of φ having the premise-sequent of the root of φ as its end-sequent.

- If the root of φ is a binary rule, then $\{1.\nu | \nu \in LKpos(\varphi_1)\} \subset LKpos(\varphi)$ and $\{2.\nu | \nu \in LKpos(\varphi_2)\} \subset LKpos(\varphi)$; where φ_1 is the sub-tree of φ having the first premise-sequent of the root of φ as its end-sequent, and φ_2 is the sub-tree of φ having the second premise-sequent of the root of φ as its end-sequent.

Definition 2.3.22 (Depth of a Position of a Proof)**.** Let ν be a position of a proof φ. Then, the *depth* of φ, $depth(\varphi)$ is defined inductively:

1. $depth(\epsilon) \doteq 0$.

2. $depth(1.\nu') \doteq 1 + depth(\nu')$.

3. $depth(2.\nu') \doteq 1 + depth(\nu')$.

Definition 2.3.23 (Sub-proofs occurring in given Positions). Let ν be a position of the **LK**-Proof φ. Then the *sub-proof* occurring in position ν is denoted $|\varphi|_\nu$. Formally, we may define the computation of $|\varphi|_\nu$ inductively:

1. $|\varphi|_\varepsilon \doteq \varphi$

2. If $\nu = 1.\nu'$ with $\nu' \in pos(\varphi_1)$, then $|\varphi|_\nu \doteq |\varphi_1|_{\nu'}$, where φ_1 is the sub-tree of φ having the first premise-sequent of the root of φ as its end-sequent.

3. If $\nu = 2.\nu'$ with $\nu' \in pos(\varphi_2)$, then $|\varphi|_\nu \doteq |\varphi_2|_{\nu'}$, where φ_2 is the sub-tree of φ having the first premise-sequent of the root of φ as its end-sequent.

Definition 2.3.24 (Sub-proof Substitution). Let φ be an **LK**-Proof. Then $\varphi_\nu[\varphi']$ denotes φ^* obtained from φ by substituting its sub-proof in position ν by φ' [4]. Alternatively we also define the following notation to express this same φ^*: $\varphi\sigma^{pp}$, where $\sigma^{pp} \doteq \{\nu/\varphi'\}$.

Definition 2.3.25 (Sequent Position in a Proof). Previously defined notations allow us to denote any sequent in a proof by first obtaining a sub-proof and then extracting the end-sequent of this sub-proof. Thus, a sequent in a Proof φ, which is the end-sequent of a sub-proof φ' in position ν can be denoted by $endseq(|\varphi|_\nu)$. Alternatively we define the following simpler notation to express this same sequent: $|\varphi|_{[\nu]}$.

Definition 2.3.26 (Sequent Substitution in a Proof). Let φ be an **LK**-Proof such that $s \doteq |\varphi|_{[\nu]}$. Then we denote by $\varphi_{[\nu]}[s']$ the proof φ' resulting from the substitution of the sequent s in position ν by the sequent s' in the proof φ. Alternatively we also define the following notation to express this same proof φ': $\varphi\sigma^{sp}$, where $\sigma^{sp} \doteq \{[\nu]/s'\}$

Definition 2.3.27 (Occurrence Position in a Proof). Previously defined notations allow us to denote an occurrence in position μ of an end-sequent s in a sub-proof φ' in position ν of a proof φ by the expression $||\varphi|_{[\nu]}|_\mu$. Alternatively we define the following simpler notation to express this same occurrence: $|\varphi|_{[\nu,\mu]}$.

Definition 2.3.28 (Occurrence Substitution in a Proof). Let φ be an **LK**-Proof such that $F \doteq |\varphi|_{[\nu,\mu]}$. Then we denote by $\varphi_{[\nu,\mu]}[F']$ the proof φ' resulting from the substitution of the occurrence F in position μ of the end-sequent of the sub-proof in position ν by the formula F'. Alternatively we also define the following notation to express this same proof φ': $\varphi\sigma^{op}$, where $\sigma^{op} \doteq \{[\nu,\mu]/F'\}$

---

[4]$\varphi_\nu[\varphi']$ will be a valid **LK**-Proof, only if $endseq(\varphi_\nu) = endseq(\varphi')$. Nevertheless, substitutions resulting in invalid proofs can still be useful as intermediary steps in proof-transformation algorithms.

Definition 2.3.29 (Sub-formula Position in a Proof). Previously defined notations allow us to denote a sub-formula in position λ of an occurrence in position μ of an end-sequent s in a sub-proof φ' in position ν of a proof φ by the expression $|||\varphi|_{[\nu]}|_\mu|_\lambda$. Alternatively we define the following simpler notation to express this same sub-formula: $|\varphi|_{[\nu,\mu,\lambda]}$.

Definition 2.3.30 (Sub-formula Substitution in a Proof). Let φ be an **LK**-Proof such that $A \doteq |\varphi|_{[\nu,\mu,\lambda]}$. Then $\varphi_{[\nu,\mu,\lambda]}[A']$ denotes the proof φ' resulting from the substitution of the sub-formula A in position λ of the occurrence in position μ of the end-sequent of the sub-proof in position ν by the formula A'. Alternatively we also define the following notation to express this same proof φ': $\varphi\sigma^{fp}$, where $\sigma^{fp} \doteq \{[\nu,\mu,\lambda]/A'\}$

2.3.5 Formula Occurrences in Proofs

Some occurrences of formulas in proofs are of particular importance, and therefore in this section we define them.

Definition 2.3.31 (Active Occurrences, Main and Auxiliary Occurrences). In the rules of Sequent Calculus some occurrences are written down explicitly with roman letters (A,B,C,...), while others occur implicitly in the context, which is written down with capital greek letters (Γ,Δ,Π,...). For a rule r in a proof φ, the formula occurrence in the conclusion sequent of r which corresponds to the formula written down explicitly in the definition of r is called *main occurrence of* r. The formulas in the premise sequents of r which correspond to formulas written down explicitly in the definition of r are called *auxiliary occurrences of* r. If a formula is a main occurrence or an auxiliary occurrence of r, then it is called *active occurrence of* r.

Definition 2.3.32 (Axiom-Occurrence, End-Occurrence, Cut-Occurrence, Terminal Occurrence). A formula occurrence in the conclusion sequent of an axiom rule is called *axiom-occurrence*. A formula occurrence in the end-sequent of a proof is called *end-occurrence*. An auxiliary occurrence of a cut-rule is called *cut-occurrence*. If an occurrence is an end-occurrence or a cut-occurrence, then it is called a *terminal occurrence*.

Definition 2.3.33 (Ancestor-Relation of Occurrences). Let A be an occurrence in a premise-sequent of a rule r in a proof φ and let C be an occurrence in the conclusion-sequent of r. A is an *immediate ancestor* of C, if A is an auxiliary occurrence of r and C is the main occurrence of r or if A occurs in the context of a premise-sequent of r and C is the corresponding occurrence in the corresponding context in the conclusion-sequent of r.

The *Ancestor-Relation* over occurrences is the transitive closure of the immediate-ancestor-relation. That is, A_1 is an ancestor of A_2 if and only if A_1 is an immediate ancestor of A_2 or there exists an immediate ancestor Z of A_2 such that A_1 is an ancestor of Z.

Table 2.1: Notations for Positions

Notation	Explanation
$|F|_\lambda$	Sub-formula in position λ of the Formula F
$|s|_\mu$	Occurrence in position μ of the sequent s
$|s|_{(\mu,\lambda)}$	Sub-formula in position λ of the occurrence in position μ of the sequent s
$|\varphi|_\nu$	Sub-proof in position ν of the proof φ
$|\varphi|_{[\nu]}$	End-sequent of the sub-proof in position ν of the proof φ
$|\varphi|_{[\nu,\mu]}$	Occurrence in position μ of the end-sequent of the sub-proof in position ν of the proof φ
$|\varphi|_{[\nu,\mu,\lambda]}$	Sub-formula in position λ of the occurrence in position μ of the end-sequent of the sub-proof in position ν of the proof φ

Definition 2.3.34 (Used Occurrences)**.** An occurrence F in a proof φ is called *used* if there is an axiom-occurrence which is an ancestor of F.

2.4 Summary of Notations for Positions and Substitutions

In the previous sections, many notations were defined to access and substitute sub-formulas, occurrences, sequents and sub-proofs in specified positions in occurrences, sequents and proofs. Tables 2.4 and 2.4 provide a summary of all the notations introduced. In this section we also give a general definition of composition of substitutions.

Definition 2.4.1 (Composition of Substitutions)**.** Let $\sigma^{\alpha\beta}$ and $\sigma^{\gamma\beta}$ be substitutions, where α,β and γ range over $\{f, o, s, p\}$ according to the possibilities displayed in table 2.4. Then $\sigma^{\alpha\beta} \circ \sigma^{\gamma\beta}$ denotes the composition of these substitutions, such that:

$$Z(\sigma^{\alpha\beta} \circ \sigma^{\gamma\beta}) \doteq (Z\sigma^{\alpha\beta})\sigma^{\gamma\beta}$$

Table 2.2: Notations for Substitutions

Notation 1	Notation 2	Explanation
$F_\lambda[A']$	$F\sigma^{fo} = F\{\lambda/A'\}$	Substitution of sub-formula in position λ of the Formula F by A'
$s_\mu[F']$	$s\sigma^{os} = s\{\mu/F'\}$	Substitution of occurrence in position μ of the sequent s by F'
$s_{(\mu,\lambda)}[A']$	$s\sigma^{fs} = s\{(\mu,\lambda)/A'\}$	Substitution of sub-formula in position λ of the occurrence in position μ of the sequent s by A'
$\varphi_\nu[\varphi']$	$\varphi\sigma^{pp} = \varphi\{\nu/\varphi'\}$	Substitution of sub-proof in position ν of the proof φ by φ'
$\varphi_{[\nu]}[s']$	$\varphi\sigma^{sp} = \varphi\{[\nu]/s'\}$	Substitution of the end-sequent of the sub-proof in position ν of the proof φ by s'
$\varphi_{[\nu,\mu]}[F']$	$\varphi\sigma^{op} = \varphi\{[\nu,\mu]/F'\}$	Substitution of the occurrence in position μ of the end-sequent of the sub-proof in position ν of the proof φ by F'
$\varphi_{[\nu,\mu,\lambda]}[A']$	$\varphi\sigma^{fp} = \varphi\{[\nu,\mu,\lambda]/A'\}$	Substitution of the sub-formula in position λ of the occurrence in position μ of the end-sequent of the sub-proof in position ν of the proof φ by A'

Chapter 3

Proof Analysis and Proof Transformations

LK-Proofs satisfying certain syntactical restrictions can be useful for different purposes. One might be interested, for example, in proofs where every sequent has formulas in negation normal form only, or in skolem normal form. Or one might want proofs in cut-normal form, in which the *cut*-rule is not used.

Given an **LK**-Proof for a sequent s, it is sometimes possible to transform this proof into another proof for s that satisfies the desired syntactical restrictions. In this chapter we define some restrictions and normal forms that are required by some of the algorithms described in subsequent chapters, and we present some theorems regarding transformations that provide **LK**-Proofs satisfying these restrictions.

3.1 Prenex Form

Intuitively, a formula in prenex form is a formula in which all quantifiers are in its beginning. One of the main purposes of the prenex form in our case, is that it is a pre-requisite for certain algorithms, as for example Gentzen's Mid-Sequent Reduction.

Any formula in First-Order Logic can be transformed to an equivalent formula in prenex form. However, this transformation reduces human-understandability, because the original positions of the quantifiers, possibly nested within propositional connectives, usually convey a reading of the formula that is closer to the way humans think.

Definition 3.1.1 (Prenex Form). A formula A is in *prenex form* if and only if it is of the form $(Q_1x_1)(Q_2x_2)\ldots(Q_nx_n)B$, for some $n \geq 0$, for $Q_1, \ldots, Q_n \in \{\forall, \exists\}$, and for B quantifier-free. A sequent s is in prenex form if and only if all its formulas are in prenex form. An **LK**-Proof is in prenex form if and only if all its sequents are in prenex form.

Theorem 3.1 (Transformation into Prenex Form)**.** For any formula A, there is a formula A' in prenex form and logically equivalent to A.

3.2 Skolemization

Skolemization is here defined as the process of removing strong quantifiers from First-Order Logic formulas or from sequents. Our definition of Skolemization generalizes more common definitions given in First-Order Logic, in order to make it suitable for Sequent Calculus and for structural skolemization.

Usually, skolemization is defined as the removal of existential quantifiers from First-Order Logic formulas, and dual-skolemization or herbrandization is defined as the removal of universal quantifiers. With these definitions, one can show that, for arbitrary F_1 and F_2 in prenex form, $skolemization(F_1) \vdash$ if and only $F_1 \vdash$ and $\vdash dualskolemization(F_2)$ if and only if $\vdash F_2$. Our generalized definition of skolemization corresponds, in the prenex case, to the removal of universal quantifiers (dual-skolemization) in the consequent and to the removal of existential quantifiers (skolemization) in the antecedent. Moreover, in the non-prenex case our definition allows structural skolemization: it does not require the non-prenex formulas or sequents to be firstly transformed to prenex normal form, as it is usually done with the more common definitions; instead it allows the undesired strong quantifiers to be removed in place.

Theorem 3.2 states that Skolemization preserves validity. Therefore, to prove that a sequent s is valid, it suffices to search for an **LK**-Proof for $Sk(s)$. This is good, because Skolemization, for our purposes, has two advantages:

1. An **LK**-Proof without quantified cut-formulas of a skolemized sequent has neither $(\forall : r)$ nor $(\exists : l)$ rules. Therefore there is no need to worry about eigen-variable conditions.

2. What appears as an eigen-variable in a non-skolemized proof, appears as a term with skolem constants and skolem functions in the corresponding skolemized proof. These terms are usually more human-understandable than just eigen-variable, because they show explicit dependencies in their expressions with skolem constants and skolem functions.

Moreover, Structural Skolemization has two advantages over Prenex Skolemization:

1. By leaving the weak quantifiers in their places, the human-readability and the human-understandability of the formula or of the sequent is better preserved.

   2. As shown in [1], Prenex Skolemization may result in a non-elementary increase in the Herbrand Complexity of an **LK**-Proof.

Below we give formal definitions and about Structural Skolemization of Formulas and Sequents. Algorithms for Structural Skolemization of **LK**-Proofs may be found in [10].

Definition 3.2.1 (Skolemization of a First-Order-Logic Formula)**.** Let $F \in F_{FOL}$. Then the *Skolemization of F, $Sk(F)$* is defined inductively as:

   1. If F does not have strong quantifiers, then $Sk(F) \doteq F$.

   2. If F has strong quantifiers and (Qy) is its first strong quantifier, then:

      (a) If (Qy) is not in the scope of weak quantifiers,
         then $Sk(F) \doteq Sk(F_{-(Qy)}\{y \leftarrow c_y\})$

      (b) If (Qy) is in the scope of the weak quantifiers $(Qx_1)(Qx_2)\ldots(Qx_n)$ appearing in this order,
         then $Sk(F) \doteq Sk(F_{-(Qy)}\{y \leftarrow f_y(x_1, x_2, \ldots, x_n)\})$

   where:

      • c_y is a constant symbol not occurring in F and is called a *skolem-constant*.

      • f_y is a function symbol not occurring in F and is called a *skolem-function*.

If the skolem-constant and skolem-function symbols are chosen deterministically in the definition of Sk, then Sk is indeed a function.

Definition 3.2.2 (Set of Skolemizations of a Formula)**.** For different choices of skolem-constant and skolem-function symbols, $Sk(F)$ produces different skolemized formulas (which are however equal modulo renaming of the skolem-constant symbols and of the skolem-function symbols). $Skolem(F)$ denotes the set of all the skolemized formulas of F:

$$Skolem(F) \doteq \{Sk_i(F)|Sk_i \text{ is a skolemization}\}$$

Definition 3.2.3 (Sequent Skolemization)**.** Let $s \doteq A_1, \ldots, A_n \vdash B_1, \ldots, B_m$ be a sequent such that $Sk(A_1 \wedge \ldots \wedge A_n \rightarrow B_1 \vee \ldots \vee B_m) = A'_1 \wedge \ldots \wedge A'_n \rightarrow B'_1 \vee \ldots \vee B'_m$. Then $Sk(s) \doteq A'_1, \ldots, A'_n \vdash B'_1, \ldots, B'_m$.

Theorem 3.2 (Validity Preservation of Skolemization)**.** Let s be a sequent. s is a valid sequent if and only if $Sk(s)$ is a valid sequent.

3.3 Herbrand Sequent

The concept of Herbrand Sequent is the core of this thesis. It allows a certain kind of reduction from First-Order Logic to propositional logic, because a Herbrand sequent for a sequent s contains sufficient instantiations for the quantified variables in s. Hence, by extracting a Herbrand Sequent from an **LK**-Proof, a summarized representation of its creative first-order content is obtained.

Herbrand Sequents, as described here, are a generalization of Herbrand Disjunctions [9] for the Sequent Calculus **LK**. Apart from the practical use described in this thesis, the concept of Herbrand Disjunction (or Herbrand Sequent) is also very important for the foundations of Logic and Mathematics, by way of Herbrand's Theorem. A concise historical and mathematical discussion of Herbrand's Theorem, as well as its relation to Gödel's Completeness Theorem, may be found in [5].

Herbrand's Theorem, by stating that every valid sequent has a Herbrand sequent, implies that it is always possible to extract Herbrand sequents from proofs, and therefore our general idea of using Herbrand sequents to summarize information about **LK**-Proofs is applicable to any **LK**-Proof, even though some of our particular algorithms may impose certain restrictions on the **LK**-Proofs that they can analyze.

A weaker form of Herbrand's Theorem, restricted to prenex sequents, is equivalent to the famous Mid-Sequent Theorem or Sharpened Hauptsatz, by Gentzen [6]. The proof of this theorem relies on a proof-transformation called Mid-Sequent Reduction, which we use as a first algorithm for the extraction of Herbrand Sequents from prenex **LK**-Proofs.

Definition 3.3.1 (Herbrand Sequents of a Sequent without Strong Quantifiers). Let $s \doteq A_1, \ldots, A_n \vdash B_1, \ldots, B_m$ be a valid sequent containing weak quantifiers only and let $A_1^0, \ldots, A_n^0, B_1^0, \ldots, B_m^0$ be its formulas without quantifiers. Furthermore, let $A_j^1, \ldots, A_j^{i_j}$ be quantifier-free substitution instances of A_j^0 (A_j^h, for $0 < h \leq i_j$, is obtained from A_j^0 by substituting its free variables by terms) and $B_k^1, \ldots, B_k^{p_k}$ be quantifier-free substitution instances of B_k^0. A valid sequent of the form

$$s' = A_1^1, \ldots, A_1^{i_1}, \ldots A_n^1, \ldots, A_n^{i_n} \vdash B_1^1, \ldots, B_1^{p_1}, \ldots, B_m^1, \ldots, B_m^{p_m}$$

is called *a Herbrand Sequent of s*.

Clearly, there are infinitely many such sequents and $H(s)$ denotes the set of all Herbrand sequents of s.

Example 3.1 (Herbrand Sequent of a Sequent). *Let $s \doteq P(0), (\forall x)(P(x) \to P(s(x))) \vdash P(s(s(0)))$. Then some Herbrand Sequents of s are:*

- $P(0), P(0) \to P(s(0)), P(s(0)) \to P(s(s(0))) \vdash P(s(s(0)))$
- $P(0), P(0), P(0) \to P(s(0)), P(s(0)) \to P(s(s(0))) \vdash P(s(s(0)))$

- $P(0), P(0) \rightarrow P(s(0)), P(s(0)) \rightarrow P(s(s(0))), P(s(s(0))) \rightarrow P(s(s(s(0)))) \vdash P(s(s(0)))$

- $\ldots$.

Definition 3.3.2 (Herbrand Sequents of a Sequent)**.** Let s be an arbitrary sequent. We define the set of its Herbrand sequents:

$$H(s) \doteq \bigcup_{s' \in Skolem(s)} H(s')$$

If $s^* \in H(s)$, then s^* is called *a Herbrand Sequent of s.*

Theorem 3.3 (Herbrand's Theorem)**.** A sequent s is valid if and only if there exists a Herbrand sequent of s.

Proof. Originally in [9], stated for Herbrand Disjunctions. Also in [5] with more modern proof calculi. □

Theorem 3.4 (Mid-Sequent Theorem or Sharpened Hauptsatz)**.** Let φ be a prenex **LK**-Proof without non-atomic cuts. Then there is an **LK**-Proof φ' of the same end-sequent such that no quantifier rule occurs above propositional and cut rules.

Proof. Original proof, for Gentzen's original Sequent Calculus **LK** and cut-free **LK**-Proofs, is available in [6]. □

3.4 Cut-Elimination

The importance of Cut-Elimination, a proof transformation invented by Gentzen [6], for the automatic analysis of mathematical proofs lies on the fact that the removal of cuts from formal **LK**-Proofs corresponds to the elimination of lemmas from informal mathematical proofs. The resulting cut-free **LK**-Proof has the property that all the formulas in its sequents are sub-formulas of formulas in the end-sequent, thus making implicit parameters explicit.

The main result, regarding cut-elimination is Gentzen's Hauptsatz or Cut-Elimination Theorem [6], which states that any **LK**-Proof can be transformed to a cut-free **LK**-Proof with the same end-sequent. However, since our variant of the Sequent Calculus **LK** allows arbitrary axiom sequents, Gentzen's Hauptsatz holds only in a weaker form, because his elimination rule for a cut on an axiom requires axiom sequents of the form $A \vdash A$. Our variant of **LK** admits transformation to proofs without non-atomic cuts, but in general atomic cuts cannot be eliminated [1].

---

[1]A simple example of an **LK**-Proof in our variant of the Sequent Calculus **LK** which does not admit a transformation to a cut-free proof is:

Although Gentzen's method for Cut-Elimination was the first one, others exist, as for example *Cut-Elimination by Resolution (CERes)* [**?**] and *Cut-Elimination by Projection* [8]. Cut-Elimination is not unique, and therefore different methods or choices within these methods may lead to new interesting proofs.

The importance of cut-elimination for this thesis lies on the fact that all algorithms require **LK**-Proofs without non-atomic cuts.

Theorem 3.5 (Weak Cut-Elimination Theorem)**.** Let φ be an **LK**-Proof. Then there exists an **LK**-Proof φ' without non-atomic cuts such that $endseq(\varphi) = endseq(\varphi')$.

Proof. A proof for the original Cut-Elimination Theorem can be found in [6]. □

$$\frac{Equal(b,a) \vdash Equal(a,b) \qquad Equal(a,b), Equal(b,c) \vdash Equal(a,c)}{Equal(b,a), Equal(b,c) \vdash Equal(a,c)} \ \text{cut}$$

This example also shows the usefulness of allowing arbitrary atomic axiom sequents as a way to express symmetry and transitivity of the predicate *Equal*.

Chapter 4

Extraction of Herbrand Sequents

Four algorithms for the Extraction of Herbrand Sequents from **LK**-Proofs are described in this chapter. Each algorithm is described in its own subsection, each organized as follows:

- Firstly, necessary notations and definitions specific for the algorithm, are introduced.

- By using all the previously defined mathematical notations and definitions, the algorithm is formally and functionally stated.

- Since the formal functional description of the algorithm is sometimes difficult to understand, a high-level and imperative pseudo-code for the algorithm is provided.

- The soundness of the algorithm is proved. This proof is divided in two lemmas: a proof that the extracted sequent is well-instantiated and a proof that the extracted sequent is valid.

- One or more examples are presented, showing how to apply the algorithm to specific **LK**-Proofs.

The algorithms may be classified on whether they require the **LK**-Proofs to be in Prenex form and on whether they need to perform transformations in order to extract the Herbrand sequent. Table 4 summarizes this classification.

Throughout this chapter, **LK**-Proofs are assumed to be free of non-atomic cuts. Thanks to theorem 3.5, a weaker form of Gentzen's Cut-Elimination Theorem, this in principle does not limit the applicability of these algorithms, because proofs could have their non-atomic cuts eliminated.

Table 4.1: Classification of the Algorithms

	Prenex LK-Proofs Required	Prenex LK-Proofs not Required
Proof-Transformations Used	Mid-Sequent Reduction	Transformation to $\mathbf{LK_A}$
Proof-Transformations not Used	Collection of Instances	Collection of Sub-formula Instances

4.1 Extraction from Prenex Proofs

The extraction of a Herbrand sequent in the case of **LK**-Proofs in prenex form is easier than in the more general case of **LK**-Proofs that are not in prenex form, and it can be done by the simpler algorithms described in the next two subsections.

By theorem 3.1, these two algorithms could in principle also be applied to non-prenex **LK**-Proofs by firstly transforming them to prenex form. However, this has disadvantages in terms of human-understandability and complexity, and therefore the two algorithms described in section 4.2 should be preferred for the general (not necessarily prenex) case. Nevertheless, the algorithms described in this section form the basis for the understanding of the more complex algorithms of the next section.

4.1.1   Extraction via Mid-Sequent Reduction

The algorithm described in this subsection is based on a mid-sequent reduction explained in [7] and similar to the one described by Gentzen in his proof of the *sharpened Hauptsatz (Mid-Sequent Theorem)* in [6]. The essential idea of the algorithm is to transform an **LK**-Proof into a proof in which no propositional-rules or cut-rules appear below quantifier-rules. Such a proof has a so-called *mid-sequent*, located between the lower part of the proof, which has the quantifier-rules, and the upper part of the proof, which has the propositional rules.

This algorithm is slightly different from the algorithm in [7], because here we allow proofs with atomic cuts, while there they require the proof to be cut-free. Unless stated otherwise, we assume **LK**-Proofs in this subsection to be free of non-atomic cuts.

Definition 4.1.1 (Degree of a Quantifier-rule). Let ρ be a quantifier rule in an **LK**-Proof φ. Then the *degree of ρ*, *degree(ρ)*, is defined as the number of propositional-rules and cut-rules in the path from ρ to the root of φ.

Definition 4.1.2 (Associated Contraction). Let φ be an **LK**-Proof containing a quantifier-rule ρ with main occurrence θ. Every contraction-rule below ρ that has an auxiliary occurrence θ' such that θ is ancestor of θ' and the only active formula occurrences this ancestor path passes through are active formula occurrences of contraction-rules, is said to be *associated with* ρ.

Example 4.1 (Associated Contractions). *Let φ be the **LK***-Proof below:*

$$
\cfrac{\cfrac{\cfrac{P(0) \vdash P(0) \qquad [\varphi_1]}{P(0), P(0) \to P(s(0)), (\forall x)(P(x) \to P(s(x)))^{(\rho_2:c_2,\rho_1:c_2)} \vdash P(s^3(0))} \to: l}{P(0), (\forall x)(P(x) \to P(s(x)))^{(\rho_3:c_2)}, (\forall x)(P(x) \to P(s(x)))^{(\rho_2:c_2,\rho_1:c_2)} \vdash P(s^3(0))} \forall: l^{[\rho_3]}}{P(0), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} c: l^{[c_2]}
$$

where φ_1 is:

$$
\cfrac{\cfrac{P(s(0)) \vdash P(s(0)) \qquad \cfrac{\cfrac{P(s^2(0)) \vdash P(s^2(0)) \qquad P(s^3(0)) \vdash P(s^3(0))}{P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0))} \to: l}{P(s^2(0)), (\forall x)(P(x) \to P(s(x)))^{(\rho_1:c_2,\rho_1:c_1)} \vdash P(s^3(0))} \forall: l^{[\rho_1]}}{\cfrac{\cfrac{P(s(0)), P(s(0)) \to P(s^2(0)), (\forall x)(P(x) \to P(s(x)))^{(\rho_1:c_2,\rho_1:c_1)} \vdash P(s^3(0))}{P(s(0)), (\forall x)(P(x) \to P(s(x)))^{(\rho_2:c_2,\rho_2:c_1)}, (\forall x)(P(x) \to P(s(x)))^{(\rho_1:c_2,\rho_1:c_1)} \vdash P(s^3(0))} \forall: l^{[\rho_2]}}{P(s(0)), (\forall x)(P(x) \to P(s(x)))^{(\rho_2:c_2,\rho_1:c_2)} \vdash P(s^3(0))} c: l^{[c_1]}} \to: l}
$$

In this proof occurrences have been labeled with $(\rho_j : c_i)$ to indicate that they belong to the ancestor path that associates c_i to ρ_j. Therefore the following associations exist in this proof:

- *the contractions associated with ρ_1 are c_1 and c_2.*

- *the contractions associated with ρ_2 are c_1 and c_2.*

- *the contraction associated with ρ_3 is c_2.*

Definition 4.1.3 (Mid-Sequent Reduction). Let φ be an **LK**-Proof. We define the transformation $\rightharpoonup_M$ pushing downwards a non-deterministically chosen quantifier-rule ρ with $degree(\rho) > 0$, according to the following cases:

1. *ρ is a $(\forall : r)$-rule*: Let τ be the first propositional or cut-rule below ρ and assume that there is no quantifier rule between ρ and τ (if this is not the case, choose the quantifier rule that is in between, instead of ρ, to push downwards).

   (a) *τ is a unary propositional-rule*: Then the subproof χ of φ with root τ has the following form:

   $$
   \begin{array}{c}
   [\psi] \\
   \dfrac{\Gamma \vdash \Delta, F\{x \leftarrow \alpha\}}{\dfrac{\Gamma \vdash \Delta, (\forall x)F}{\dfrac{\Gamma' \vdash \Delta', (\forall x)F}{\Gamma'' \vdash \Delta'', (\forall x)F} \, c : *, w : *^{[\delta]}} \forall : r^{[\rho]}} * : *^{[\tau]}
   \end{array}
   $$

   The premise of τ contains $(\forall x)F$ (as a descendant of the main occurrence of ρ) because neither contractions nor weakenings can remove it. The propositional rule τ does not have this $(\forall x)F$ as auxiliary formula (for assuming it has would lead to contradiction, because the main formula of τ wouldn't be in prenex form). Hence the conclusion of τ also contains this $(\forall x)F$.

   Let δ' be the contractions in δ that are associated with ρ and let Π be the multi-set that contains the formula $(\forall x)F$ exactly n times where n is the number of contractions in δ'.

   We define $\chi \rightharpoonup_M \chi'$ with χ' as shown below:

   $$
   \begin{array}{c}
   [\psi] \\
   \dfrac{\Gamma \vdash \Delta, F\{x \leftarrow \alpha\}}{\dfrac{\Gamma' \vdash \Delta', \Pi, F\{x \leftarrow \alpha\}}{\dfrac{\Gamma'' \vdash \Delta'', \Pi, F\{x \leftarrow \alpha\}}{\dfrac{\Gamma'' \vdash \Delta'', \Pi, (\forall x)F}{\Gamma'' \vdash \Delta'', (\forall x)F} \, c : *^{[\delta']}} \forall : r^{[\rho]}} * : *^{[\tau]}} \, c : *, w : *^{[\delta \setminus \delta']}
   \end{array}
   $$

   (b) *τ is a binary rule and ρ is in the left branch*: Then the subproof χ of φ with root τ has the following form:

   $$
   \begin{array}{c}
   [\psi] \\
   \dfrac{\dfrac{\dfrac{\Gamma \vdash \Delta, F\{x \leftarrow \alpha\}}{\Gamma \vdash \Delta, (\forall x)F} \forall : r^{[\rho]}}{\Gamma' \vdash \Delta', (\forall x)F} \, c : *, w : *^{[\delta]} \qquad \Gamma'_1 \vdash \Delta'_1}{\Gamma'', \Gamma''_1 \vdash \Delta'', \Delta''_1, (\forall x)F} * : *^{[\tau]}
   \end{array}
   $$

The left premise of τ contains $(\forall x)F$ (as a descendant of the main occurrence of ρ) because neither contractions nor weakenings can remove it. The propositional rule τ does not have this $(\forall x)F$ as auxiliary formula (for assuming it has would lead to contradiction, because the main formula of τ wouldn't be in prenex form). Hence the conclusion of τ also contains this $(\forall x)F$.

Let δ' be the contractions in δ that are associated with ρ and let Π be the multi-set that contains the formula $(\forall x)F$ exactly n times where n is the number of contractions in δ'.

We define $\chi \rightharpoonup_M \chi'$ with χ' as shown below:

$$
\cfrac{
  \cfrac{
    \cfrac{
      \cfrac{
        [\psi] \\
        \Gamma \vdash \Delta, F\{x \leftarrow \alpha\}
      }{\Gamma' \vdash \Delta', \Pi, F\{x \leftarrow \alpha\}} \; c : *, w : *^{[\delta \backslash \delta']}
      \qquad
      \Gamma_1' \vdash \Delta_1'
    }{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', \Pi, F\{x \leftarrow \alpha\}} \; * : *^{[\tau]}
  }{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', \Pi, (\forall x)F} \; \forall : r^{[\rho]}
}{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', (\forall x)F} \; c : *^{[\delta']}
$$

(c) *τ is a binary rule and ρ is in the right branch:* Then the subproof χ of φ with root τ has the following form:

$$
\cfrac{
  \Gamma_1' \vdash \Delta_1'
  \qquad
  \cfrac{
    \cfrac{
      \cfrac{
        [\psi] \\
        \Gamma \vdash \Delta, F\{x \leftarrow \alpha\}
      }{\Gamma \vdash \Delta, (\forall x)F} \; \forall : r^{[\rho]}
    }{\Gamma' \vdash \Delta', (\forall x)F} \; c : *, w : *^{[\delta]}
  }{}
}{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', (\forall x)F} \; * : *^{[\tau]}
$$

The right premise of τ contains $(\forall x)F$ (as a descendant of the main occurrence of ρ) because neither contractions nor weakenings can remove it. The propositional rule τ does not have this $(\forall x)F$ as auxiliary formula (for assuming it has would lead to contradiction, because the main formula of τ wouldn't be in prenex form). Hence the conclusion of τ also contains this $(\forall x)F$.

Let δ' be the contractions in δ that are associated with ρ and let Π be the multi-set that contains the formula $(\forall x)F$ exactly n times where n is the number of contractions in δ'.

We define $\chi \rightharpoonup_M \chi'$ with χ' as shown below:

$$
\cfrac{
  \cfrac{
    \cfrac{
      \Gamma_1' \vdash \Delta_1'
      \qquad
      \cfrac{
        [\psi] \\
        \Gamma \vdash \Delta, F\{x \leftarrow \alpha\}
      }{\Gamma' \vdash \Delta', \Pi, F\{x \leftarrow \alpha\}} \; c : *, w : *^{[\delta \backslash \delta']}
    }{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', \Pi, F\{x \leftarrow \alpha\}} \; * : *^{[\tau]}
  }{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', \Pi, (\forall x)F} \; \forall : r^{[\rho]}
}{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', (\forall x)F} \; c : *^{[\delta']}
$$

The eigen-variable condition of ρ in χ' is fulfilled because α does not occur in Γ, Δ, Γ', Δ'. In particular, α is not added by a weakening in δ because the proof is regular.

2. *ρ is a $(\exists : r)$-rule*: Let τ be the first propositional or cut-rule below ρ and assume that there is no quantifier rule between ρ and τ (if this is not the case, choose the quantifier rule that is in between, instead of ρ, to push downwards).

   (a) *τ is a unary propositional-rule*: Then the subproof χ of φ with root τ has the following form:

$$
\begin{array}{c}
[\psi] \\
\dfrac{\Gamma \vdash \Delta, F\{x \leftarrow t\}}{\dfrac{\dfrac{\Gamma \vdash \Delta, (\exists x)F}{\Gamma' \vdash \Delta', (\exists x)F}\ c:*,w:*^{[\delta]}}{\Gamma'' \vdash \Delta'', (\exists x)F}\ *:*^{[\tau]}}\ \exists:r^{[\rho]}
\end{array}
$$

   The premise of τ contains $(\exists x)F$ (as a descendant of the main occurrence of ρ) because neither contractions nor weakenings can remove it. The propositional rule τ does not have this $(\exists x)F$ as auxiliary formula (for assuming it has would lead to contradiction, because the main formula of τ wouldn't be in prenex form). Hence the conclusion of τ also contains this $(\exists x)F$.

   Let δ' be the contractions in δ that are associated with ρ and let Π be the multi-set that contains the formula $(\exists x)F$ exactly n times where n is the number of contractions in δ'.

   We define $\chi \rightharpoonup_M \chi'$ with χ' as shown below:

$$
\begin{array}{c}
[\psi] \\
\dfrac{\Gamma \vdash \Delta, F\{x \leftarrow t\}}{\dfrac{\dfrac{\dfrac{\dfrac{\Gamma' \vdash \Delta', \Pi, F\{x \leftarrow t\}}{\Gamma'' \vdash \Delta'', \Pi, F\{x \leftarrow t\}}\ *:*^{[\tau]}}{\Gamma'' \vdash \Delta'', \Pi, (\exists x)F}\ \exists:r^{[\rho]}}{\Gamma'' \vdash \Delta'', (\exists x)F}\ c:*^{[\delta']}}{}}{}\ c:*,w:*^{[\delta \setminus \delta']}
\end{array}
$$

   (b) *τ is a binary rule and ρ is in the left branch*: Then the subproof χ of φ with root τ has the following form:

$$
\begin{array}{c}
[\psi] \\
\dfrac{\dfrac{\dfrac{\Gamma \vdash \Delta, F\{x \leftarrow t\}}{\Gamma \vdash \Delta, (\exists x)F}\ \exists:r^{[\rho]}}{\Gamma' \vdash \Delta', (\exists x)F}\ c:*,w:*^{[\delta]} \qquad \Gamma_1' \vdash \Delta_1'}{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', (\exists x)F}\ *:*^{[\tau]}
\end{array}
$$

The left premise of τ contains $(\exists x)F$ (as a descendant of the main occurrence of ρ) because neither contractions nor weakenings can remove it. The propositional rule τ does not have this $(\exists x)F$ as auxiliary formula (for assuming it has would lead to contradiction, because the main formula of τ wouldn't be in prenex form). Hence the conclusion of τ also contains this $(\exists x)F$.

Let δ' be the contractions in δ that are associated with ρ and let Π be the multi-set that contains the formula $(\exists x)F$ exactly n times where n is the number of contractions in δ'.

We define $\chi \rightharpoonup_M \chi'$ with χ' as shown below:

$$
\cfrac{
\cfrac{
\cfrac{
\cfrac{
\begin{array}{c}[\psi]\\ \Gamma \vdash \Delta, F\{x \leftarrow t\}\end{array}
}{\Gamma' \vdash \Delta', \Pi, F\{x \leftarrow t\}} \; c:*, w:*^{[\delta \backslash \delta']} \qquad \Gamma_1' \vdash \Delta_1'
}{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', \Pi, F\{x \leftarrow t\}} \; *:*^{[\tau]}
}{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', \Pi, (\exists x)F} \; \exists:r^{[\rho]}
}{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', (\exists x)F} \; c:*^{[\delta']}
$$

(c) *τ is a binary rule and ρ is in the right branch*: Then the subproof χ of φ with root τ has the following form:

$$
\cfrac{
\Gamma_1' \vdash \Delta_1' \qquad
\cfrac{
\cfrac{
\cfrac{\begin{array}{c}[\psi]\\ \Gamma \vdash \Delta, F\{x \leftarrow t\}\end{array}}{\Gamma \vdash \Delta, (\exists x)F} \; \exists:r^{[\rho]}
}{\Gamma' \vdash \Delta', (\exists x)F} \; c:*, w:*^{[\delta]}
}{}
}{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', (\exists x)F} \; *:*^{[\tau]}
$$

The right premise of τ contains $(\exists x)F$ (as a descendant of the main occurrence of ρ) because neither contractions nor weakenings can remove it. The propositional rule τ does not have this $(\exists x)F$ as auxiliary formula (for assuming it has would lead to contradiction, because the main formula of τ wouldn't be in prenex form). Hence the conclusion of τ also contains this $(\exists x)F$.

Let δ' be the contractions in δ that are associated with ρ and let Π be the multi-set that contains the formula $(\exists x)F$ exactly n times where n is the number of contractions in δ'.

We define $\chi \rightharpoonup_M \chi'$ with χ' as shown below:

$$
\cfrac{
\cfrac{
\cfrac{
\Gamma_1' \vdash \Delta_1' \qquad
\cfrac{\begin{array}{c}[\psi]\\ \Gamma \vdash \Delta, F\{x \leftarrow t\}\end{array}}{\Gamma' \vdash \Delta', \Pi, F\{x \leftarrow t\}} \; c:*, w:*^{[\delta \backslash \delta']}
}{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', \Pi, F\{x \leftarrow t\}} \; *:*^{[\tau]}
}{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', \Pi, (\exists x)F} \; \exists:r^{[\rho]}
}{\Gamma'', \Gamma_1'' \vdash \Delta'', \Delta_1'', (\exists x)F} \; c:*^{[\delta']}
$$

3. *ρ is a $(\forall : l)$-rule*: Let τ be the first propositional or cut-rule below ρ and assume that there is no quantifier rule between ρ and τ (if this is not the case, choose the quantifier rule that is in between, instead of ρ, to push downwards).

   (a) *τ is a unary propositional-rule*: Then the subproof χ of φ with root τ has the following form:

$$
\frac{\dfrac{\dfrac{[\psi]}{\Gamma, F\{x \leftarrow t\} \vdash \Delta}}{\dfrac{\Gamma, (\forall x)F \vdash \Delta}{\dfrac{\Gamma', (\forall x)F \vdash \Delta'}{\Gamma'', (\forall x)F \vdash \Delta''}}}{}
$$

$$
\frac{\Gamma, F\{x \leftarrow t\} \vdash \Delta}{\Gamma, (\forall x)F \vdash \Delta} \forall : l^{[\rho]}
$$
$$
\frac{}{\Gamma', (\forall x)F \vdash \Delta'} c : *, w : *^{[\delta]}
$$
$$
\frac{}{\Gamma'', (\forall x)F \vdash \Delta''} * : *^{[\tau]}
$$

The premise of τ contains $(\forall x)F$ (as a descendant of the main occurrence of ρ) because neither contractions nor weakenings can remove it. The propositional rule τ does not have this $(\forall x)F$ as auxiliary formula (for assuming it has would lead to contradiction, because the main formula of τ wouldn't be in prenex form). Hence the conclusion of τ also contains this $(\forall x)F$.

Let δ' be the contractions in δ that are associated with ρ and let Π be the multi-set that contains the formula $(\forall x)F$ exactly n times where n is the number of contractions in δ'.

We define $\chi \twoheadrightarrow_M \chi'$ with χ' as shown below:

$$
\frac{[\psi]}{\Gamma, F\{x \leftarrow t\} \vdash \Delta}
$$
$$
\frac{}{\Gamma', F\{x \leftarrow t\}, \Pi \vdash \Delta'} c : *, w : *^{[\delta \backslash \delta']}
$$
$$
\frac{}{\Gamma'', F\{x \leftarrow t\}, \Pi \vdash \Delta''} * : *^{[\tau]}
$$
$$
\frac{}{\Gamma'', (\forall x)F, \Pi \vdash \Delta''} \forall : l^{[\rho]}
$$
$$
\frac{}{\Gamma'', (\forall x)F \vdash \Delta''} c : *^{[\delta']}
$$

   (b) *τ is a binary rule and ρ is in the left branch*: Then the subproof χ of φ with root τ has the following form:

$$
\frac{\dfrac{[\psi]}{\Gamma, F\{x \leftarrow t\} \vdash \Delta}}{\dfrac{\Gamma, (\forall x)F \vdash \Delta}{\Gamma', (\forall x)F \vdash \Delta'}} \quad \Gamma_1' \vdash \Delta_1'
$$
$$
\frac{\Gamma, F\{x \leftarrow t\} \vdash \Delta}{\Gamma, (\forall x)F \vdash \Delta} \forall : l^{[\rho]}
$$
$$
\frac{}{\Gamma', (\forall x)F \vdash \Delta'} c : *, w : *^{[\delta]} \qquad \Gamma_1' \vdash \Delta_1'
$$
$$
\frac{}{\Gamma'', \Gamma_1'', (\forall x)F \vdash \Delta'', \Delta_1''} * : *^{[\tau]}
$$

The left premise of τ contains $(\forall x)F$ (as a descendant of the main occurrence of ρ) because neither contractions nor weakenings can remove it. The propositional rule τ does not have this $(\forall x)F$ as auxiliary formula (for assuming it has would lead to contradiction,

because the main formula of τ wouldn't be in prenex form). Hence the conclusion of τ also contains this $(\forall x)F$.

Let δ' be the contractions in δ that are associated with ρ and let Π be the multi-set that contains the formula $(\forall x)F$ exactly n times where n is the number of contractions in δ'.

We define $\chi \rightharpoonup_M \chi'$ with χ' as shown below:

$$
\cfrac{
  \cfrac{
    \cfrac{
      \cfrac{
        \begin{array}{c} [\psi] \\ \Gamma, F\{x \leftarrow t\} \vdash \Delta \end{array}
      }{\Gamma', F\{x \leftarrow t\}, \Pi \vdash \Delta'} \; c : *, w : *^{[\delta \backslash \delta']}
      \qquad
      \Gamma'_1 \vdash \Delta'_1
    }{\Gamma'', \Gamma''_1, F\{x \leftarrow t\}, \Pi \vdash \Delta'', \Delta''_1} \; * : *^{[\tau]}
  }{\Gamma'', \Gamma''_1, (\forall x)F, \Pi \vdash \Delta'', \Delta''_1} \; \forall : l^{[\rho]}
}{\Gamma'', \Gamma''_1, (\forall x)F \vdash \Delta'', \Delta''_1} \; c : *^{[\delta']}
$$

(c) τ *is a binary rule and* ρ *is in the right branch*: Then the subproof χ of φ with root τ has the following form:

$$
\cfrac{
  \Gamma'_1 \vdash \Delta'_1
  \qquad
  \cfrac{
    \cfrac{
      \begin{array}{c} [\psi] \\ \Gamma, F\{x \leftarrow t\} \vdash \Delta \end{array}
    }{\Gamma, (\forall x)F \vdash \Delta} \; \forall : l^{[\rho]}
    \;\; \raisebox{-1ex}{$$}
  }{\Gamma', (\forall x)F \vdash \Delta'} \; c : *, w : *^{[\delta]}
}{\Gamma'', \Gamma''_1, (\forall x)F \vdash \Delta'', \Delta''_1} \; * : *^{[\tau]}
$$

The right premise of τ contains $(\forall x)F$ (as a descendant of the main occurrence of ρ) because neither contractions nor weakenings can remove it. The propositional rule τ does not have this $(\forall x)F$ as auxiliary formula (for assuming it has would lead to contradiction, because the main formula of τ wouldn't be in prenex form). Hence the conclusion of τ also contains this $(\forall x)F$.

Let δ' be the contractions in δ that are associated with ρ and let Π be the multi-set that contains the formula $(\forall x)F$ exactly n times where n is the number of contractions in δ'.

We define $\chi \rightharpoonup_M \chi'$ with χ' as shown below:

$$
\cfrac{
  \cfrac{
    \cfrac{
      \Gamma'_1 \vdash \Delta'_1
      \qquad
      \cfrac{
        \begin{array}{c} [\psi] \\ \Gamma, F\{x \leftarrow t\} \vdash \Delta \end{array}
      }{\Gamma', F\{x \leftarrow t\}, \Pi \vdash \Delta'} \; c : *, w : *^{[\delta \backslash \delta']}
    }{\Gamma'', \Gamma''_1, F\{x \leftarrow t\}, \Pi \vdash \Delta'', \Delta''_1} \; * : *^{[\tau]}
  }{\Gamma'', \Gamma''_1, (\forall x)F, \Pi \vdash \Delta'', \Delta''_1} \; \forall : l^{[\rho]}
}{\Gamma'', \Gamma''_1, (\forall x)F \vdash \Delta'', \Delta''_1} \; c : *^{[\delta']}
$$

4. ρ *is a* $(\exists : l)$-*rule*: Let τ be the first propositional or cut-rule below ρ and assume that there is no quantifier rule between ρ and τ (if this is not the case, choose the quantifier rule that is in between, instead of ρ, to push downwards).

(a) τ *is a unary propositional-rule*: Then the subproof χ of φ with root τ has the following form:

$$\frac{\dfrac{[\psi]}{\Gamma, F\{x \leftarrow \alpha\} \vdash \Delta}}{\dfrac{\dfrac{\Gamma, (\exists x)F \vdash \Delta}{\Gamma', (\exists x)F \vdash \Delta'} \; c : *, w : *^{[\delta]}}{\Gamma'', (\exists x)F \vdash \Delta''} \; * : *^{[\tau]}} \; \exists : l^{[\rho]}$$

The premise of τ contains $(\exists x)F$ (as a descendant of the main occurrence of ρ) because neither contractions nor weakenings can remove it. The propositional rule τ does not have this $(\exists x)F$ as auxiliary formula (for assuming it has would lead to contradiction, because the main formula of τ wouldn't be in prenex form). Hence the conclusion of τ also contains this $(\exists x)F$.

Let δ' be the contractions in δ that are associated with ρ and let Π be the multi-set that contains the formula $(\exists x)F$ exactly n times where n is the number of contractions in δ'.

We define $\chi \rightarrowtail_M \chi'$ with χ' as shown below:

$$\frac{[\psi]}{\dfrac{\dfrac{\dfrac{\dfrac{\Gamma, F\{x \leftarrow \alpha\} \vdash \Delta}{\Gamma', F\{x \leftarrow \alpha\}, \Pi \vdash \Delta'} \; c : *, w : *^{[\delta \backslash \delta']}}{\Gamma'', F\{x \leftarrow \alpha\}, \Pi \vdash \Delta''} \; * : *^{[\tau]}}{\Gamma'', (\exists x)F, \Pi \vdash \Delta''} \; \exists : l^{[\rho]}}{\Gamma'', (\exists x)F \vdash \Delta''} \; c : *^{[\delta']}}$$

(b) τ *is a binary rule and ρ is in the left branch*: Then the subproof χ of φ with root τ has the following form:

$$\frac{\dfrac{\dfrac{[\psi]}{\Gamma, F\{x \leftarrow \alpha\} \vdash \Delta}}{\dfrac{\Gamma, (\exists x)F \vdash \Delta}{\Gamma', (\exists x)F \vdash \Delta'} \; c : *, w : *^{[\delta]}} \; \exists : l^{[\rho]} \qquad \Gamma'_1 \vdash \Delta'_1}{\Gamma'', \Gamma''_1, (\exists x)F \vdash \Delta'', \Delta''_1} \; * : *^{[\tau]}$$

The left premise of τ contains $(\exists x)F$ (as a descendant of the main occurrence of ρ) because neither contractions nor weakenings can remove it. The propositional rule τ does not have this $(\exists x)F$ as auxiliary formula (for assuming it has would lead to contradiction, because the main formula of τ wouldn't be in prenex form). Hence the conclusion of τ also contains this $(\exists x)F$.

Let δ' be the contractions in δ that are associated with ρ and let Π be the multi-set that contains the formula $(\exists x)F$ exactly n times where n is the number of contractions in δ'.

We define $\chi \rightharpoonup_M \chi'$ with χ' as shown below:

$$\cfrac{\cfrac{\cfrac{[\psi]}{\Gamma, F\{x \leftarrow \alpha\} \vdash \Delta}}{\Gamma', F\{x \leftarrow \alpha\}, \Pi \vdash \Delta'} \; c : *, w : *^{[\delta \backslash \delta']} \qquad \Gamma_1' \vdash \Delta_1'}{\cfrac{\cfrac{\Gamma'', \Gamma_1'', F\{x \leftarrow \alpha\}, \Pi \vdash \Delta'', \Delta_1''}{\Gamma'', \Gamma_1'', (\exists x)F, \Pi \vdash \Delta'', \Delta_1''} \; \exists : l^{[\rho]}}{\Gamma'', \Gamma_1'', (\exists x)F \vdash \Delta'', \Delta_1''} \; c : *^{[\delta']}} \; * : *^{[\tau]}}$$

(c) τ *is a binary rule and* ρ *is in the right branch*: Then the subproof χ of φ with root τ has the following form:

$$\cfrac{\Gamma_1' \vdash \Delta_1' \qquad \cfrac{\cfrac{\cfrac{[\psi]}{\Gamma, F\{x \leftarrow \alpha\} \vdash \Delta}}{\Gamma, (\exists x)F \vdash \Delta} \; \exists : l^{[\rho]}}{\Gamma', (\exists x)F \vdash \Delta'} \; c : *, w : *^{[\delta]}}{\Gamma'', \Gamma_1'', (\exists x)F \vdash \Delta'', \Delta_1''} \; * : *^{[\tau]}$$

The right premise of τ contains $(\exists x)F$ (as a descendant of the main occurrence of ρ) because neither contractions nor weakenings can remove it. The propositional rule τ does not have this $(\exists x)F$ as auxiliary formula (for assuming it has would lead to contradiction, because the main formula of τ wouldn't be in prenex form). Hence the conclusion of τ also contains this $(\exists x)F$.

Let δ' be the contractions in δ that are associated with ρ and let Π be the multi-set that contains the formula $(\exists x)F$ exactly n times where n is the number of contractions in δ'.

We define $\chi \rightharpoonup_M \chi'$ with χ' as shown below:

$$\cfrac{\cfrac{\cfrac{\Gamma_1' \vdash \Delta_1' \qquad \cfrac{\cfrac{[\psi]}{\Gamma, F\{x \leftarrow \alpha\} \vdash \Delta}}{\Gamma', F\{x \leftarrow \alpha\}, \Pi \vdash \Delta'} \; c : *, w : *^{[\delta \backslash \delta']}}{\Gamma'', \Gamma_1'', F\{x \leftarrow \alpha\}, \Pi \vdash \Delta'', \Delta_1''} \; * : *^{[\tau]}}{\Gamma'', \Gamma_1'', (\exists x)F, \Pi \vdash \Delta'', \Delta_1''} \; \exists : l^{[\rho]}}{\Gamma'', \Gamma_1'', (\exists x)F \vdash \Delta'', \Delta_1''} \; c : *^{[\delta']}$$

The eigen-variable condition of ρ in χ' is fulfilled because α does not occur in Γ, Δ, Γ', Δ'. In particular, α is not added by a weakening in δ because the proof is regular.

We define $\varphi \rightharpoonup_M \varphi'$ with $\varphi' \doteq \varphi_\nu[\chi']$, where ν is the position of χ in φ. In other words, $\rightharpoonup_M$ is compatibility-closed. $\rightharpoonup_M^*$ denotes the reflexive and transitive closure of $\rightharpoonup_M$.

Definition 4.1.4 (Mid-Sequent Normal Form). an **LK**-Proof φ is in *mid-sequent normal form*, $M - NF$, if and only if there is no **LK**-Proof φ' such that $\varphi \rightharpoonup_M \varphi'$.

Theorem 4.1 (Existence of Mid-Sequent Normal Form). For every **LK**-Proof φ, there exists an **LK**-Proof φ' such that $\varphi \rightharpoonup^*_M \varphi'$ and φ' is in Mid-Sequent Normal Form.

Proof. This follows from Gentzen's Sharpened Hauptsatz (Mid Sequent Theorem), in [6]. □

Theorem 4.2 (Splitting of a Prenex Proof in Mid-Sequent Normal Form). An **LK**-Proof in Mid-Sequent Normal Form of a prenex end-sequent and without non-atomic cuts has $degree(\rho) = 0$ for any quantifier-rule ρ. This means that there is no propositional or cut-rule below a quantifier-rule. Such a proof can thus be split in two parts: an upper part containing only propositional-rules and structural-rules; and a lower part containing only quantifier-rules, contraction-rules and weakening-rules. However this splitting is not unique, because we can split the proof at any position between the lowest propositional-rule or cut-rule and the highest quantifier-rule. Between these two rules there may be several contractions and weakenings.

Theorem 4.3 (Non-Confluency of Mid-Sequent Reduction). Mid-Sequent Reduction is not confluent. For an **LK**-Proof φ, there may be more than one **LK**-Proof φ' in mid-sequent normal form such that $\varphi \rightharpoonup_M \varphi'$.

Algorithm 4.1 (Herbrand Sequent Extraction via Proof Transformation (Mid-Sequent Reduction)). We may extract a Herbrand sequent from an **LK**-Proof φ with end-sequent in prenex form by executing the following steps:

1. Find (by executing mid-sequent reductions) φ' such that $\varphi \rightharpoonup^*_M \varphi'$ and φ' is in mid-sequent normal form.

2. Let ρ be the lowest axiom-rule, propositional-rule or cut-rule (ρ in in the path between the end-sequent and any other axiom-rule, propositional-rule or cut-rule). And let s be the conclusion sequent of ρ. Then compute and output $H_M(\varphi) \doteq set(U(s))$, where $U(s)$ is the sequent containing only the used occurrences of s and $set(s')$ is the set-normalized sequent of s' (s' with duplicate occurrences removed).

Algorithm 4.1.1: $H_M(\varphi)$

$\varphi' \leftarrow \varphi$
while (φ' not in Mid-Sequent Normal Form)
   do $\varphi' \leftarrow \textsc{MidSequentReduction}(\varphi')$
$\rho \leftarrow$ lowest axiom, propositional or cut rule
$s \leftarrow$ conclusion sequent of ρ
$s_H \leftarrow \textsc{UsednessNormalization}(s)$
$s_H \leftarrow \textsc{SetNormalization}(s_H)$
return (s_H)

Lemma 4.1 (Extracted Sequent Well-Instantiated). Let φ be an **LK**-Proof and let its end-sequent be $|\varphi|_{[\epsilon]} = A_1, \ldots, A_n \vdash B_1, \ldots, B_m$. $A_1^0, \ldots, A_n^0, B_1^0, \ldots, B_m^0$ are its formulas without quantifiers. Then the sequent extracted by algorithm 4.1 has the form:

$$H_M(\varphi) = A_1^1, \ldots, A_1^{i_1}, \ldots A_n^1, \ldots, A_n^{i_n} \vdash B_1^1, \ldots, B_1^{p_1}, \ldots, B_m^1, \ldots, B_m^{p_m}$$

where: $A_j^1, \ldots, A_j^{i_j}$ are quantifier-free substitution instances of A_j^0 and $B_k^1, \ldots, B_k^{p_k}$ are quantifier-free substitution instances of B_k^0.

Proof. Let φ' and s be a normal form of φ and its mid-sequent as specified in algorithm 4.1. Since above s we don't have quantifier-rules, none of the occurrences in s can have quantified variables. Therefore, all occurrences of s are quantifier-free formulas. Furthermore, these occurrences must be substitution instances of the formulas in $|\varphi|_{[\epsilon]}$, because below s we have only quantifier-rules, contractions and weakenings. $H_M(\varphi) = set(U(s))$ is a subsequent of s, therefore its occurrences are quantifier-free substitution instances of the occurrences in $|\varphi|_{[\epsilon]}$. $\qquad\square$

Lemma 4.2 (Validity of the Extracted Sequent). Let φ be an **LK**-Proof. Then $H_M(\varphi)$ is valid.

Proof. As specified in algorithm 4.1, let φ' be a normal form of φ and ρ its lowest propositional-rule, axiom-rule or cut-rule. Let s be the conclusion sequent of ρ, and let ν be the position of the subproof of φ' which has ρ as its root. Then we can construct the following **LK**-Proof for $H_M(\varphi)$, thus demonstrating its validity:

$$\frac{\begin{array}{c} [|\varphi'|_\nu] \\ s \end{array}}{H_M(\varphi)} \, w : *, c : *$$

$\qquad\square$

Theorem 4.4 (Soundness of Algorithm 4.1). Let φ be an **LK**-Proof. Then the sequent extracted by algorithm 4.1, $H_M(\varphi)$, is a Herbrand sequent.

Proof. Lemmas 4.1 and 4.2 show that the extracted sequent satisfies the requirements in definition 3.3.2, therefore it is a Herbrand sequent. □

Theorem 4.5 (Determinacy of $H_M(\varphi)$). The output of algorithm 4.1, $H_M(\varphi)$ is uniquely defined.

Proof. This is not trivial, because the non-confluency of mid-sequent reduction, implying that there may be different proofs φ' in mid-sequent normal form for φ, means that the first step of algorithm 4.1 is non-deterministic. However, even though step 1 may result in different proofs φ', step 2 gives the same result for any of these possible proofs φ' in mid-sequent normal form. This is a corollary of theorem 4.6, which shows the equality of the results obtained by algorithms 4.1 and 4.2. Since algorithm 4.2 is deterministic, the sequent extracted by algorithm 4.1 must also be uniquely defined. □

Definition 4.1.5 (Herbrand Sequent of an **LK**-Proof in Prenex Form). The determinacy of the output of algorithm 4.1 (as expressed in theorem **??**) and theorem 4.8 together allow us to define *The Herbrand Sequent of an* **LK**-*Proof φ with End-Sequent in Prenex Form*, $H_P(\varphi)$, as the very result of the algorithm, $H_M(\varphi)$.

Example 4.2 (Mid-Sequent Reduction of an **LK**-Proof with Prenex End-sequent to a $M - NF$ and Extraction of its Herbrand Sequent). *Let φ be the* **LK**-*Proof below:*

$$
\cfrac{\cfrac{P(0) \vdash P(0) \qquad \cfrac{[\varphi_*]}{P(s(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))}}{P(0), P(0) \to P(s(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \to: l}{\cfrac{P(0), (\forall x)(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))}{P(0), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} c : l} \forall : l
$$

where φ_* is:

$$
\cfrac{P(s(0)) \vdash P(s(0)) \qquad \cfrac{\cfrac{\cfrac{P(s^2(0)) \vdash P(s^2(0)) \qquad P(s^3(0)) \vdash P(s^3(0))}{P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0))} \to: l}{P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \forall : l^{[1]}}{} }{}
$$

$$
\cfrac{\cfrac{P(s(0)) \vdash P(s(0)) \qquad \cfrac{\cfrac{\cfrac{P(s^2(0)) \vdash P(s^2(0)) \quad P(s^3(0)) \vdash P(s^3(0))}{P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0))} \to: l}{P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \forall : l^{[1]}}{P(s(0)), P(s(0)) \to P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \to: l}{\cfrac{P(s(0)), (\forall x)(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))}{P(s(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} c : l} \forall : l^{[2]}
$$

φ has two quantifier rules (marked with numbers [1] and [2] that appear above propositional rules. Therefore φ is not in $M - NF$. By applying 3 mid-sequent reductions, as demonstrated below, we can produce an **LK**-Proof φ' which is in $M - NF$. Firstly, we permute the quantifier rule [1] with the $(\rightarrow: l)$-rule immediately below it:

$$
\begin{array}{c}
\vdots \\
\cfrac{
  P(s(0)) \vdash P(s(0)) \qquad
  \cfrac{
    \cfrac{P(s^2(0)), P(s^2(0)) \rightarrow P(s^3(0)) \vdash P(s^3(0))}{P(s^2(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \; \forall : l^{[1]}
  }{}
}{P(s(0)), P(s(0)) \rightarrow P(s^2(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \;\; \rightarrow: l \\
\vdots
\end{array}
$$

$$\downarrow_M$$

$$
\begin{array}{c}
\vdots \\
\cfrac{
  \cfrac{
    P(s(0)) \vdash P(s(0)) \qquad P(s^2(0)), P(s^2(0)) \rightarrow P(s^3(0)) \vdash P(s^3(0))
  }{P(s(0)), P(s(0)) \rightarrow P(s^2(0)), P(s^2(0)) \rightarrow P(s^3(0)) \vdash P(s^3(0))} \; \rightarrow: l
}{P(s(0)), P(s(0)) \rightarrow P(s^2(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \;\; \forall : l^{[1]} \\
\vdots
\end{array}
$$

Secondly, we permute the quantifier rule [2] with the first $(\rightarrow: l)$-rule below it:

$$
\begin{array}{c}
\vdots \\
\cfrac{
  P(0) \vdash P(0) \qquad
  \cfrac{
    \cfrac{
      \cfrac{P(s(0)), P(s(0)) \rightarrow P(s^2(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))}{P(s(0)), (\forall x)(P(x) \rightarrow P(s(x))), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \; \forall : l^{[2]}
    }{P(s(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \; c : l
  }{}
}{P(0), P(0) \rightarrow P(s(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \;\; \rightarrow: l \\
\vdots
\end{array}
$$

$$\downarrow_M$$

$$
\begin{array}{c}
\vdots \\
\cfrac{
  \cfrac{
    \cfrac{
      P(0) \vdash P(0) \qquad P(s(0)), P(s(0 \rightarrow P(s^2(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))
    }{P(0), P(0) \rightarrow P(s(0)), P(s(0)) \rightarrow P(s^2(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \; \rightarrow: l
  }{P(0), P(0) \rightarrow P(s(0)), (\forall x)(P(x) \rightarrow P(s(x))), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \; \forall : l^{[2]}
}{P(0), P(0) \rightarrow P(s(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \;\; c : l \\
\vdots
\end{array}
$$

After these two mid-sequent reductions, our transformed proof is currently:

$$
\cfrac{
  \cfrac{
    \cfrac{
      \cfrac{
        \cfrac{
          P(0) \vdash P(0) \qquad
          \cfrac{
            P(s(0)) \vdash P(s(0)) \qquad
            \cfrac{
              P(s^2(0)) \vdash P(s^2(0)) \qquad P(s^3(0)) \vdash P(s^3(0))
            }{ P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0)) } \;\to:l
          }{
            \cfrac{ P(s(0)), P(s(0)) \to P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0)) }{ P(s(0)), P(s(0) \to P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) } \;\forall:l^{[1]}
          } \;\to:l
        }{ P(0), P(0) \to P(s(0)), P(s(0)) \to P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) } \;\to:l
      }{ P(0), P(0) \to P(s(0)), (\forall x)(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) } \;\forall:l^{[2]}
    }{ P(0), P(0) \to P(s(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) } \;c:l
  }{ P(0), (\forall x)(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) } \;\forall:l
}{ P(0), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) } \;c:l
$$

Quantifier rule number [1] is still above propositional rules, therefore another mid-sequent reduction is necessary:

$$
\vdots
$$

$$
\cfrac{
  P(0) \vdash P(0) \qquad
  \cfrac{
    P(s(0)), P(s(0) \to P(s^2(0)), P(s^2(0) \to P(s^3(0)) \vdash P(s^3(0))
  }{ P(s^2(0)), P(s(0) \to P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) } \;\forall:l^{[1]}
}{ P(0), P(0) \to P(s(0)), P(s(0)) \to P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) } \;\to:l
$$

$$
\vdots
$$

$$
\downarrow_M
$$

$$
\vdots
$$

$$
\cfrac{
  P(0) \vdash P(0) \qquad
  \cfrac{
    P(s(0)), P(s(0) \to P(s^2(0)), P(s^2(0) \to P(s^3(0)) \vdash P(s^3(0))
  }{
    \cfrac{ P(0), P(0) \to P(s(0)), P(s(0)) \to P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0)) }{ P(0), P(0) \to P(s(0)), P(s(0)) \to P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) } \;\forall:l^{[1]}
  } \;\to:l
}{ }
$$

$$
\vdots
$$

This last mid-sequent reduction produces an **LK***-Proof* φ' *in* $M - NF$:

$$
\cfrac{
  \cfrac{
    \cfrac{
      \cfrac{
        \cfrac{
          P(0) \vdash P(0) \qquad
          \cfrac{
            P(s(0)) \vdash P(s(0)) \qquad
            \cfrac{
              P(s^2(0)) \vdash P(s^2(0)) \qquad P(s^3(0)) \vdash P(s^3(0))
            }{ P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0)) } \;\to:l
          }{ P(s(0)), P(s(0)) \to P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0)) } \;\to:l
        }{ P(0), P(0) \to P(s(0)), P(s(0)) \to P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0)) } \;\to:l
      }{ P(0), P(0) \to P(s(0)), P(s(0)) \to P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) } \;\forall:l^{[1]}
    }{ P(0), P(0) \to P(s(0)), (\forall x)(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) } \;\forall:l^{[2]}
  }{ P(0), P(0) \to P(s(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) } \;c:l
}{
  \cfrac{ P(0), (\forall x)(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) }{ P(0), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0)) }
}
$$

We can now extract the Herbrand sequent of φ*, which in this case is simply the conclusion sequent of the lowest* ($\to:l$)*-rule of* φ':

$$
H_M(\varphi) = H_M(\varphi') = \begin{pmatrix} P(0), \\ P(0) \to P(s(0)), \\ P(s(0)) \to P(s^2(0)), \\ P(s^2(0)) \to P(s^3(0)) \end{pmatrix} \vdash P(s^3(0))
$$

4.1.2   Extraction via Collection of Instances

Algorithm 4.1 extracts a Herbrand Sequent by performing several mid-sequent reductions that transform the original proof. This extensive transformation is quite a waste of computational effort if we are not interested in the transformed proof, but only in a Herbrand Sequent for the end-sequent of the original proof.

It turns out that it is possible to compute the same Herbrand Sequent just by analyzing the original proof, instead of transforming it. In this subsection we explain this algorithm, which was firstly described in [7].

The idea is to notice that quantifier-free substitution instances in occurrences of $H_P(\varphi)$ defined in 4.1.5 are necessarily the auxiliary occurrences of some quantifier-rules in φ. Therefore we can compute $H_P(\varphi)$ just by analyzing φ, collecting its appropriate auxiliary occurrences and then constructing a sequent by composing the quantifier-free subsequent with the sequent formed from these collected auxiliary occurrences. In other words, we remove the quantified occurrences of the end-sequent and in their place we put their substitution instances given by the auxiliary occurrences of the quantifier-rules.

By showing that the result of applying algorithm 4.2 is exactly the same as the result of applying algorithm 4.1, the soundness of algorithm 4.2 follows trivially from the already proved soundness of algorithm 4.1. Furthermore, since algorithm 4.2 is deterministic, it follows that the non-determinism of algorithm 4.1 is irrelevant, since its result is necessarily unique and equal to the result of algorithm 4.2.

Definition 4.1.6 (Set of Quantifier Rules). $Q(\varphi)$ is the set of all quantifier-rules of the **LK**-Proof φ.

Definition 4.1.7 (Sets of Used Auxiliary Occurrences). Let R be a set of rules. With $\mathcal{A}_A(R)$ we denote the set of used auxiliary occurrences in the antecedent of the rules in R. With $\mathcal{A}_C(R)$ we denote the set of used auxiliary occurrences in the consequent of the rules in R.

Definition 4.1.8 (Used Subsequent). Let s be a sequent. With $U(s)$ we denote the subsequent of s obtained by removing from s all occurrences that are not used.

Definition 4.1.9 (Quantifier-free Subsequent). Let s be a sequent. With $\mathcal{D}(s)$ we denote the subsequent of s obtained by removing from s all occurrences that contain quantifiers.

Algorithm 4.2 (Herbrand Sequent Extraction via Proof Analysis (Instance Collection)). Let φ be an **LK**-Proof with end-sequent s in prenex form. Then a Herbrand sequent of the end-sequent of φ may be extracted from φ by computing:

$$H_{IC} \doteq set(s_P(\varphi) \circ Q_P(\varphi))$$

where:

$$s_P(\varphi) \doteq \mathcal{D}(U(s))$$

$$Q_P(\varphi) \doteq \mathcal{D}(seq(\mathcal{A}_A(Q(\varphi)), \mathcal{A}_A(Q(\varphi))))$$

Algorithm 4.1.2: $H_{IC}(\varphi)$

$s \leftarrow$ end-sequent of φ
$s_P \leftarrow$ USEDNESSNORMALIZATION(s)
$s_P \leftarrow$ QUANTIFIERFREESUBSEQUENT(s_P)
$Q \leftarrow$ QUANTIFIERRULES(φ)
$U_A \leftarrow$ ANTECEDENTUSEDAUXILIARYOCCURRENCES(Q)
$U_C \leftarrow$ CONSEQUENTUSEDAUXILIARYOCCURRENCES(Q)
$Q_P \leftarrow$ SEQUENT(U_A, U_C)
$Q_P \leftarrow$ QUANTIFIERFREESUBSEQUENT(Q_P)
$s_H \leftarrow s_P \circ Q_P$
$s_H \leftarrow$ SETNORMALIZATION(s_H)
return (s_H)

Theorem 4.6 (Equality of Results obtained via Algorithms 4.2 and 4.1)**.**
Let φ be an **LK**-Proof with end-sequent in prenex form. Then:

$$H_M(\varphi) = H_{IC}(\varphi)$$

Proof. The proof below is adapted from [7].

Let φ be a prenex **LK**-Proof. Let φ^* be a mid-sequent normal form of φ computed non-deterministically by algorithm 4.1.

We proceed by induction on the length n of the Mid-Sequent Reduction sequence of φ to φ^*.

1. For $n = 0$ we have $\varphi = \varphi^*$ and φ in Mid-Sequent normal form. We show

$$H_M(\varphi) = set(s_P(\varphi) \circ Q_P(\varphi))$$

   by induction on the size of the quantifier part.

   Let ρ be the lowest propositional-rule, axiom-rule or cut-rule and let s be its conclusion sequent. Then $H_M(\varphi) = set(U(s))$. We can partition φ as $\begin{array}{c} \varphi_p \\ \varphi_q \end{array}$ where φ_p ends with ρ and φ_q consists only of quantifier rules, weakenings and contractions (which are all unary). We proceed by induction on m, the number of rules in φ_q:

   If $m = 0$ then $Q_P(\varphi) = \emptyset$, $s_P = U(s)$ and therefore

$$H_{IC}(\varphi) = set(s_P(\varphi) \circ Q_P(\varphi)) = set(U(s)) = H_M(\varphi)$$

If $m > 0$, let $\varphi_q = \dfrac{\varphi'_q}{\rho'}$.

By Induction Hypothesis we know for $\varphi' = \dfrac{\varphi_p}{\varphi'_q}$ that $H_M(\varphi') = set(s_P(\varphi') \circ Q_P(\varphi'))$. But $H_M(\varphi) = H_M(\varphi_p) = H_M(\varphi')$ and we also have $set(s_P(\varphi') \circ Q_P(\varphi')) = set(s_P(\varphi) \circ Q_P(\varphi))$ because if ρ' is a quantifier rule with an unused auxiliary occurrence or a contraction or a weakening, then $set(s_P(\varphi)) = set(s_P(\varphi'))$ and $Q_P(\varphi) = Q_P(\varphi')$, and if ρ' is a quantifier rule with a used auxiliary formula occurrence F, then F moves from $s_P(\varphi')$ into $Q_P(\varphi)$.

2. For the induction step ($n > 0$) we show that: if $\varphi \rightharpoonup_M \varphi'$, then:

    (a) $Q_P(\varphi) = Q_P(\varphi')$
    (b) $s_P(\varphi) = s_P(\varphi')$.

    That $s_P(\varphi) = s_P(\varphi')$ is the case comes from the fact that mid-sequent reduction does not modify the end-sequent and preserves usedness.

    To prove that $Q_P(\varphi) = Q_P(\varphi')$ we observe that no quantifier rules are added, nor removed. Moreover, mid-sequent reduction also does not add nor remove weakenings and the ancestor relation in the modified parts is not modified. So

    $$seq(\mathcal{A}_A(Q(\varphi)), \mathcal{A}_C(Q(\varphi))) = seq(\mathcal{A}_A(Q(\varphi')), \mathcal{A}_C(Q(\varphi')))$$

    , and thus $Q_P(\varphi) = Q_P(\varphi')$.

    $\square$

Lemma 4.3 (Extracted Sequent Well-Instantiated)**.** Let φ be an **LK**-Proof and let its end-sequent be
$|\varphi|_{[\epsilon]} = A_1, \dots, A_n \vdash B_1, \dots, B_m$. $A_1^0, \dots, A_n^0, B_1^0, \dots, B_m^0$ are its formulas without quantifiers. Then the sequent extracted by algorithm 4.2 has the form:

$$H_{IC}(\varphi) = A_1^1, \dots, A_1^{i_1}, \dots A_n^1, \dots, A_n^{i_n} \vdash B_1^1, \dots, B_1^{p_1}, \dots, B_m^1, \dots, B_m^{p_m}$$

where: $A_j^1, \dots, A_j^{i_j}$ are quantifier-free substitution instances of A_j^0 and $B_k^1, \dots, B_k^{p_k}$ are quantifier-free substitution instances of B_k^0.

Proof. By theorem 4.6 we have $H_{IC}(\varphi) = H_M(\varphi)$, and by lemma 4.1, $H_M(\varphi)$ has the desired form. Therefore $H_{IC}(\varphi)$ also has the desired form.
$\square$

Lemma 4.4 (Validity of the Extracted Sequent)**.** Let φ be an **LK**-Proof. Then $H_IC(\varphi)$ is valid.

Proof. By theorem 4.6 we have $H_{IC}(\varphi) = H_M(\varphi)$, and by lemma 4.2, $H_M(\varphi)$ is valid. Therefore $H_{IC}(\varphi)$ is also valid.

$$\frac{\dfrac{[|\varphi'|_\nu]}{s}}{H_M(\varphi)} \, w : *, c : *$$

$\square$

Theorem 4.7 (Soundness of Algorithm 4.2)**.** Let φ be an **LK**-Proof. Then the sequent extracted by algorithm 4.2, $H_IC(\varphi)$, is a Herbrand sequent.

Proof. Lemmas 4.3 and 4.4 show that the extracted sequent satisfies the requirements in definition 3.3.2, therefore it is a Herbrand sequent. $\square$

Example 4.3 (Extraction of the Herbrand Sequent of an **LK**-Proof of a Prenex End-Sequent without Proof Transformation)**.** *Let φ be the **LK**-Proof in example 4.2:*

$$\small
\cfrac{
P(0) \vdash P(0) \quad
\cfrac{
P(s(0)) \vdash P(s(0)) \quad
\cfrac{
\cfrac{
\cfrac{
P(s^2(0)) \vdash P(s^2(0)) \quad P(s^3(0)) \vdash P(s^3(0))
}{P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0))} \, {\to}{:}\, l
}{P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \, \forall : l^{[1]}
}{
\cfrac{
\cfrac{P(s(0)), P(s(0)) \to P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))}{P(s(0)), (\forall x)(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \, \forall : l^{[2]}
}{P(s(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \, c : l
} \, {\to}{:}\, l
}{
\cfrac{
\cfrac{P(0), P(0) \to P(s(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))}{P(0), (\forall x)(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \, \forall : l
}{P(0), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \, c : l
}
$$

The set $Q(\varphi)$ of quantifier rules of φ has the following rules:

$$\frac{P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0))}{P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \, \forall : l^{[1]}$$

$$\frac{P(s(0)), P(s(0)) \to P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))}{P(s(0)), (\forall x)(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \, \forall : l^{[2]}$$

$$\frac{P(0), P(0) \to P(s(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))}{P(0), (\forall x)(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \, \forall : l$$

Its set of used auxiliary antecedent occurrences is:

$$\mathcal{A}_A(Q(\varphi)) = \left\{ \begin{array}{l} P(0) \to P(s(0)), \\ P(s(0)) \to P(s^2(0)), \\ P(s^2(0)) \to P(s^3(0)) \end{array} \right\}$$

Its set of used auxiliary consequent occurrences is:

$$\mathcal{A}_C(Q(\varphi)) = \{\}$$

The sequent formed from these used occurrences is:

$$seq(\mathcal{A}_A(Q(\varphi))), \mathcal{A}_C(Q(\varphi)))) = \begin{pmatrix} P(0) \rightarrow P(s(0)), \\ P(s(0)) \rightarrow P(s^2(0)), \\ P(s^2(0)) \rightarrow P(s^3(0)) \end{pmatrix} \vdash$$

Since all the formulas in $seq(\mathcal{A}_A(Q(\varphi))), \mathcal{A}_C(Q(\varphi))))$ are quantifier-free, we have:

$$\begin{aligned} Q_P(\varphi) &= \mathcal{D}(seq(\mathcal{A}_A(Q(\varphi))), \mathcal{A}_C(Q(\varphi)))))) \\ &= seq(\mathcal{A}_A(Q(\varphi))), \mathcal{A}_C(Q(\varphi)))) \\ &= \begin{pmatrix} P(0) \rightarrow P(s(0)), \\ P(s(0)) \rightarrow P(s^2(0)), \\ P(s^2(0)) \rightarrow P(s^3(0)) \end{pmatrix} \vdash \end{aligned}$$

Furthermore:

$$\begin{aligned} s_P(\varphi) &= \mathcal{D}(U(s)) \\ &= P(0) \vdash P(s^3(0)) \end{aligned}$$

And hence, we have:

$$\begin{aligned} H(\varphi) &= set(s_P(\varphi) \circ Q_P(\varphi)) \\ &= \begin{pmatrix} P(0), \\ P(0) \rightarrow P(s(0)), \\ P(s(0)) \rightarrow P(s^2(0)), \\ P(s^2(0)) \rightarrow P(s^3(0) \end{pmatrix} \vdash P(s^3(0)) \end{aligned}$$

4.2 Extraction from Non-Prenex Proofs

When the end-sequent is not in Prenex form, mid-sequent reduction clearly cannot be used to extract a Herbrand sequent, because rules referring to quantifiers nested under propositional connectives must necessarily occur above the rules referring to these propositional connectives, and thus mid-sequent reduction will not be capable of pushing down these quantifier rules until they are all below all the propositional rules. Example 4.4 shows this.

Algorithm 4.2 also cannot be used, because the collected instances are all included without distinction with depth 0 in the constructed Herbrand sequent, while in the Non-Prenex case, these instances should be included in the nested position from which they originate.

In principle, those two algorithms could still be used, if the non-prenex **LK**-Proofs were firstly transformed to prenex form, but this would decrease human-understandability, because it is easier for humans to understand formulas and proofs when the quantifiers are in their original places than in the beginning of the formulas as a result of prenexification.

Therefore, we must design new algorithms to extract Herbrand sequents from **LK**-Proofs of end-sequents in non-prenex form. Here we describe two such algorithms. The first one, *Extraction via Proof Transformation to Quantifier-free* **LK$_\mathbf{A}$**, was described in [1] and involves the definition of an auxiliary formula-structure (array-formula) and modifications in the Sequent Calculus to deal with this structure. The second algorithm, *Extraction via Collection of Sub-Formula Instances*, generalizes the algorithm of *Extraction via Collection of Instances* for the case of non-prenex end-sequents by using the same formula-structure defined for the first algorithm.

Example 4.4 (End-sequent in non-prenex form and an **LK**-Proof for it)**.** *Let the end-sequent be $P(0) \wedge (\forall x)(P(x) \to P(s(x)) \vdash P(s^2(0))$, which is not in prenex-form. Any* **LK***-proof of this end-sequent will have a propositional rule $\wedge : l$ below the quantifier rules $\forall : l$, which are responsible for inserting the $\forall$-quantifier in copies (obtained by contraction) of the sub-formula $\forall x(P(x) \to P(s(x)))$. One example of such a proof is the* **LK***-Proof φ shown below. A Herbrand sequent cannot be obtained by applying the mid-sequent theorem in this case, because the quantifier rules cannot be pushed down until they are brought under the $\wedge : l$ rule.*

$$
\cfrac{
\cfrac{
\cfrac{
\cfrac{
\begin{array}{c} [\varphi'] \\ P(0), P(0) \to P(s(0)), P(s(0)) \to P(s^2(0)) \vdash P(s^2(0)) \end{array}
}{P(0), P(0) \to P(s(0)), (\forall x)(P(x) \to P(s(x)) \vdash P(s^2(0))} \ \forall : l
}{P(0), \forall x(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x)) \vdash P(s^2(0))} \ \forall : l
}{P(0), \forall x(P(x) \to P(s(x)) \vdash P(s^2(0))} \ c : l
}{P(0) \wedge (\forall x)(P(x) \to P(s(x)) \vdash P(s^2(0))} \ \wedge : l
$$

4.2.1 Extraction via Proof Transformation to Quantifier-free LK$_\mathbf{A}$

The algorithm described in this subsection is restricted to **LK**-Proofs of end-sequents with weak quantifiers only. The reason for this is that the proof transformations used in this algorithm would violate the eigen-variable conditions. This is not a limitation for the algorithm, since end-sequents may have their strong quantifiers eliminated via skolemization. Therefore, unless stated otherwise, we assume that sequents don't have strong quantifiers.

The method basically consists of transforming the **LK**-Proof into a quantifier free proof in a modified version of Sequent Calculus called **LK$_\mathbf{A}$**, which admits sequents containing so-called *array-formulas*. Then the end-sequent of this **LK$_\mathbf{A}$**-Proof is transformed back to a sequent without array-formulas. As proved in theorem 4.8, this final sequent is indeed a Herbrand sequent of the end-sequent of the original **LK**-Proof, and thus the algorithm here described is sound. Furthermore, the soundness and determinism of the algorithm allow us to associate a unique Herbrand sequent for each $LK - Proof$ of an end-sequent in non-prenex form with weak quantifiers only (Definition 4.2.4).

Definition 4.2.1 (Array Formula)**.** An *array formula* is defined inductively as:

1. First-Order-Logic Formulas are Array Formulas

2. If $A_1, A_2, \ldots, A_n$ are Array Formulas, then $\langle A_1, A_2, \ldots, A_n \rangle$

3. If A, B are Array Formulas, then $\neg A, A \vee B, A \wedge B, A \rightarrow B$ are Array Formulas.

Definition 4.2.2 (Sequent Calculus **LK$_\mathbf{A}$**)**.** If $A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m$ are Array Formulas, then $A_1, A_2, \ldots, A_n, \vdash B_1, B_2, \ldots, B_m$ is called an Array Sequent.

The Sequent Calculus **LK$_\mathbf{A}$** is the Sequent Calculus **LK** with addition of the following rules:

$$\frac{\Delta, A, \Gamma, B, \Pi \vdash \Lambda}{\Delta, \langle A, B \rangle, \Gamma, \Pi \vdash \Lambda} \ \langle \rangle : l \qquad \frac{\Lambda \vdash \Delta, A, \Gamma, B, \Pi}{\Lambda \vdash \Delta, \langle A, B \rangle, \Gamma, \Pi} \ \langle \rangle : l$$

Algorithm 4.3 (Ψ: Transformation of **LK**-Proofs into Quantifier-free **LK$_\mathbf{A}$**-Proofs)**.** We define a mapping Ψ which transforms an arbitrary **LK**-Proof φ into an **LK$_\mathbf{A}$**-Proof without quantifiers by executing the following two steps:

1. We construct φ' by omitting all quantifier-rules.

2. φ' is generally not an **LK**-Proof, because its remaining contraction-rules are not necessarily correct anymore. Therefore we replace these "false" contraction-rules (starting from above downwards) by $\langle \rangle : l$ and $\langle \rangle : r$ rules. If

$$\frac{\Gamma, A_1, \Delta, A_2, \Pi \vdash \Lambda}{\Gamma, A_3, \Delta, \Pi \vdash \Lambda} \; c : l$$

occurs in φ' and $A_1 \neq A_2$, then replace it by:

$$\frac{\Gamma, A_1, \Delta, A_2, \Pi \vdash \Lambda}{\Gamma, \langle A_1, A_2 \rangle, \Delta, \Pi \vdash \Lambda} \; \langle \rangle : l$$

Analogously, if:

$$\frac{\Lambda \vdash \Gamma, A_1, \Delta, A_2, \Pi}{\Lambda \vdash \Gamma, A_3, \Delta, \Pi} \; c : r$$

occurs in φ' and $A_1 \neq A_2$, then replace it by:

$$\frac{\Lambda \vdash \Gamma, A_1, \Delta, A_2, \Pi}{\Lambda \vdash \Gamma, \langle A_1, A_2 \rangle, \Delta, \Pi} \; \langle \rangle : r$$

$\Psi(\varphi)$ is the quantifier-free **LK$_A$**-Proof obtained by executing the two steps above.

Example 4.5 (Transforming an **LK**-Proof with quantifiers into a quantifier-free **LK$_A$**-proof). *Let φ be the **LK**-proof shown in example 4.4. Firstly we omit all quantifier rules and obtain the following **LK**-Proof φ_c with false contractions:*

$$[\varphi']$$

$$\frac{\dfrac{P(0), P(0) \rightarrow P(s(0)), P(s(0)) \rightarrow P(s^2(0)) \vdash P(s^2(0))}{P(0), (\forall x)(P(x) \rightarrow P(s(x)) \vdash P(s^2(0))} \; c : l}{P(0) \wedge (\forall x)(P(x) \rightarrow P(s(x)) \rightarrow P(s^2(0)) \vdash P(s^2(0))} \; \wedge : l$$

*Then we replace the false contraction rule by an array formation rule, thus obtaining the **LK$_A$**-Proof $\Psi(\varphi)$.*

$$[\varphi']$$

$$\frac{\dfrac{P(0), P(0) \rightarrow P(s(0)), P(s(0)) \rightarrow P(s^2(0)) \vdash P(s^2(0))}{P(0), \langle P(0) \rightarrow P(s(0)), P(s(0)) \rightarrow P(s^2(0)) \rangle \vdash P(s^2(0))} \; \langle \rangle : l}{P(0) \wedge \langle P(0) \rightarrow P(s(0)), P(s(0)) \rightarrow P(s^2(0)) \rangle \vdash P(s^2(0))} \; \wedge : l$$

Definition 4.2.3 (Φ: Mapping from Array Formulas to sequences of FOL–Formulas). We define a mapping Φ which transforms array formulas and sequents into First-Order Logic formulas and sequents. In other words, Φ eliminates $\langle \ldots \rangle$ and can be defined inductively by:

1. If $A \in F_F OL$, then $\Phi(A) \doteq A$

2. $\Phi(\langle A, B \rangle) \doteq \Phi(A), \Phi(B)$

3. If $\Phi(A) = A_1, \ldots, A_n$, then $\Phi(\neg A) \doteq \neg A_1, \ldots, \neg A_n$

4. If $\Phi(A) = A_1, \ldots, A_n$ and $\Phi(B) = B_1, \ldots, B_m$, then $\Phi(A \circ B) \doteq A_1 \circ B_1, \ldots, A_1 \circ B_m, \ldots, A_n \circ B_1, \ldots, A_n \circ B_m$, for $\circ \in \{\wedge, \vee, \rightarrow\}$

5. If $\Phi(A) = A_1, \ldots, A_n$, then $\Phi((Qx)(A)) \doteq (Qx)(A_1), \ldots, (Qx)(A_n)$, for $Q \in \{\forall, \exists\}$.

6. $\Phi(A_1, \ldots, A_n \vdash B_1, \ldots, B_m) \doteq \Phi(A_1), \ldots, \Phi(A_n) \vdash \Phi(B_1), \ldots, \Phi(B_m)$

Example 4.6 (Mapping from Array Formulas to sequences of FOL-Formulas). *Let* $\Psi(\varphi)$ *be the* **LK$_A$***-Proof in example 4.5. Then, its end-sequent mapped to FOL-Formulas is:*

$$\Phi(|\Psi(\varphi)|_{[\epsilon]}) = \begin{pmatrix} P(0) \wedge P(0) \rightarrow P(s(0)), \\ P(0) \wedge P(s(0)) \rightarrow P(s^2(0)) \end{pmatrix} \vdash P(s^2(0))$$

Algorithm 4.4 (Herbrand Sequent Extraction via Proof Transformation to Quantifier-free **LK$_A$**). Let φ be an **LK**-Proof. A Herbrand Sequent of $|\varphi|_{[\epsilon]}$ can be obtained by computing:

$$H_{LK_A}(\varphi) \doteq \Phi(|\Psi(\varphi)|_{[\epsilon]})$$

Algorithm 4.2.1: $H_{LK_A}(\varphi)$

$\varphi' \leftarrow \Psi(\varphi)$
$s_H \leftarrow$ end-sequent of φ'
$s_H \leftarrow \Phi(s_H)$
return (s_H)

Lemma 4.5 (Extracted Sequent Well-Instantiated). Let φ be an **LK**-Proof and let its end-sequent be $|\varphi|_{[\epsilon]} = A_1, \ldots, A_n \vdash B_1, \ldots, B_m$.
$A_1^0, \ldots, A_n^0, B_1^0, \ldots, B_m^0$ are its formulas without quantifiers. Then the sequent extracted by algorithm 4.4 has the form:

$$H_{LK_A}(\varphi) = A_1^1, \ldots, A_1^{i_1}, \ldots A_n^1, \ldots, A_n^{i_n} \vdash B_1^1, \ldots, B_1^{p_1}, \ldots, B_m^1, \ldots, B_m^{p_m}$$

where: $A_j^1, \ldots, A_j^{i_j}$ are quantifier-free substitution instances of A_j^0 and $B_k^1, \ldots, B_k^{p_k}$ are quantifier-free substitution instances of B_k^0.

Proof. By definition of Φ ([1]). $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\square$

Algorithm 4.5 (Φ_P: Transformation of an **LK$_\mathbf{A}$**-Proof into an **LK**-Proof)**.**
We define a mapping Φ_P which transforms an arbitrary **LK$_\mathbf{A}$**-Proof φ into an
LK-Proof without quantifiers by replacing (starting from above downwards)
the $\langle \rangle : l$ and $\langle \rangle : r$ rules. If

$$\frac{\Gamma, A_1, \Delta, A_2, \Pi \vdash \Lambda}{\Gamma, \langle A_1, A_2 \rangle, \Delta, \Pi \vdash \Lambda} \ \langle \rangle : l$$

occurs in φ', then replace it by:

$$\frac{\Gamma, A_1, \Delta, A_2, \Pi \vdash \Lambda}{\Gamma, (A_1 \wedge A_2), \Delta, \Pi \vdash \Lambda} \ \wedge : l$$

Analogously, if:

$$\frac{\Lambda \vdash \Gamma, A_1, \Delta, A_2, \Pi}{\Lambda \vdash \Gamma, \langle A_1, A_2 \rangle, \Delta, \Pi} \ \langle \rangle : r$$

occurs in φ' and $A_1 \neq A_2$, then replace it by:

$$\frac{\Lambda \vdash \Gamma, A_1, \Delta, A_2, \Pi}{\Lambda \vdash \Gamma, (A_1 \vee A_2), \Delta, \Pi} \ \vee : r$$

$\Phi_P(\varphi)$ is the quantifier **LK**-Proof obtained by executing the procedure
above.

Lemma 4.6 (Equivalence of $H_{LK_A}(\varphi)$ and $|\Phi_P(\Psi(\varphi))|_{[\epsilon]}$)**.** Let φ be an **LK**-
Proof. Then the sequents $H_{LK_A}(\varphi)$ and $|\Phi_P(\Psi(\varphi))|_{[\epsilon]}$ are logically equiva-
lent.

Proof. Let $s \doteq |\varphi|_{[\varepsilon]}$ be the end-sequent of φ. Let $s' \doteq |\Psi(\varphi)|_{[\varepsilon]}$ be the end-
sequent of $\Psi(\varphi)$. Let $s^* \doteq |\Phi_P(\Psi(\varphi))|_{[\varepsilon]}$ be the end-sequent of $\Phi_P(\Psi(\varphi))$.
By the definition of Φ_P, we note that:

- If $\langle A_1, A_2 \rangle$ occurs positively in s', then $(A_1 \vee A_2)$ occurs in the corre-
  sponding position in s^*;

- If $\langle A_1, A_2 \rangle$ occurs negatively in s', then $(A_1 \wedge A_2)$ occurs in the corre-
  sponding position in s^*;

Now we prove, by structural induction, that for each formula A in posi-
tion μ in s', the sequence of formulas $\Phi(A)$ is logically equivalent to $|s^*|_\mu$:

- If $A \in F_{FOL}$, then $\Phi(A) = A = |s^*|_\mu$.

- If $A = \langle A_1, A_2 \rangle$ occurs positively, then $|s^*|_\mu = (A_1 \vee A_2)$. Furthermore
  $s^* = \Gamma \vdash \Delta, (A_1 \vee A_2), \Pi \sim \Gamma \vdash \Delta, A_1, A_2, \Pi$. Hence $|s^*|_\mu \sim A_1^+, A_2^+ = \Phi(A)$.

- If $A = \langle A_1, A_2 \rangle$ occurs negatively, then $|s^*|_\mu = (A_1 \wedge A_2)$. Furthermore $s^* = \Pi, (A_1 \wedge A_2), \Gamma \vdash \Delta \sim \Pi, A_1, A_2, \Gamma \vdash \Delta$. Hence $|s^*|_\mu \sim A_1^-, A_2^- = \Phi(A)$.

- If A is of the form $\neg B$ and B occurs positively and $\Phi(B) = B_1, \ldots, B_n$, then by induction hypothesis, we have $|s^*|_{(\mu,1)} \sim \Phi(B)^+ \sim ((\ldots (B_1 \vee B_2) \ldots) \vee B_n)$. Hence $|s^*|_\mu \sim ((\ldots (\neg B_1 \wedge \neg B_2) \ldots) \wedge \neg B_n)$. And since $\Phi(A) = \neg B_1, \ldots, \neg B_n$, we have: $|s^*|_\mu \sim \Phi(A)^-$.

- If A is of the form $\neg B$ and B occurs negatively and $\Phi(B) = B_1, \ldots, B_n$, then by induction hypothesis, we have $|s^*|_{(\mu,1)} \sim \Phi(B)^- \sim ((\ldots (B_1 \wedge B_2) \ldots) \wedge B_n)$. Hence $|s^*|_\mu \sim ((\ldots (\neg B_1 \vee \neg B_2) \ldots) \vee \neg B_n)$. And since $\Phi(A) = \neg B_1, \ldots, \neg B_n$, we have: $|s^*|_\mu \sim \Phi(A)^+$.

- If A occurs positively and it is of the form $B \circ C$ with $\circ \in \{\wedge, \vee\}$ and $\Phi(B) = B_1, \ldots, B_n$ and $\Phi(C) = C_1, \ldots, C_m$, then by induction hypothesis we have $|s^*|_{(\mu,1)} \sim \Phi(B)^+ \sim ((\ldots (B_1 \vee B_2) \ldots \vee B_n)$ and $|s^*|_{(\mu,2)} \sim \Phi(C)^+ \sim ((\ldots (C_1 \vee C_2) \ldots \vee C_n)$. Hence, by distributivity of $\circ$ with respect to $\vee$, $|s^*|_\mu \sim ((\ldots ((\ldots (B_1 \circ C_1) \vee \ldots) \vee (B_1 \circ C_m)) \ldots \vee ((\ldots (B_n \circ C_1) \vee \ldots) \vee (B_n \circ C_m))$. Therefore, $|s^*|_\mu \sim \Phi(A)^+ = B_1 \circ C_1, \ldots, B_1 \circ C_m, \ldots, B_n \circ C_1, \ldots, B_n \circ C_m$.

- If A occurs negatively and it is of the form $B \circ C$ with $\circ \in \{\wedge, \vee\}$ and $\Phi(B) = B_1, \ldots, B_n$ and $\Phi(C) = C_1, \ldots, C_m$, then by induction hypothesis we have $|s^*|_{(\mu,1)} \sim \Phi(B)^- \sim ((\ldots (B_1 \wedge B_2) \ldots \wedge B_n)$ and $|s^*|_{(\mu,2)} \sim \Phi(C)^- \sim ((\ldots (C_1 \wedge C_2) \ldots \wedge C_n)$. Hence, by distributivity of $\circ$ with respect to $\wedge$, $|s^*|_\mu \sim ((\ldots ((\ldots (B_1 \circ C_1) \wedge \ldots) \wedge (B_1 \circ C_m)) \ldots \wedge ((\ldots (B_n \circ C_1) \wedge \ldots) \wedge (B_n \circ C_m))$. Therefore, $|s^*|_\mu \sim \Phi(A)^- = B_1 \circ C_1, \ldots, B_1 \circ C_m, \ldots, B_n \circ C_1, \ldots, B_n \circ C_m$.

- If A occurs positively and it is of the form $B \rightarrow C$, and $\Phi(B) = B_1, \ldots, B_n$ and $\Phi(C) = C_1, \ldots, C_m$, then by induction hypothesis we have $|s^*|_{(\mu,1)} \sim \Phi(B)^- \sim ((\ldots (B_1 \wedge B_2) \ldots \wedge B_n)$ and $|s^*|_{(\mu,2)} \sim \Phi(C)^+ \sim ((\ldots (C_1 \vee C_2) \ldots \vee C_n)$. Hence, by distributivity of $\rightarrow$ with respect to $\vee$ and $\wedge$, $|s^*|_\mu \sim ((\ldots ((\ldots (B_1 \rightarrow C_1) \vee \ldots) \vee (B_1 \rightarrow C_m)) \ldots \vee ((\ldots (B_n \rightarrow C_1) \vee \ldots) \vee (B_n \rightarrow C_m))$. Therefore, $|s^*|_\mu \sim \Phi(A)^+ = B_1 \rightarrow C_1, \ldots, B_1 \rightarrow C_m, \ldots, B_n \rightarrow C_1, \ldots, B_n \rightarrow C_m$.

- If A occurs negatively and it is of the form $B \rightarrow C$, and $\Phi(B) = B_1, \ldots, B_n$ and $\Phi(C) = C_1, \ldots, C_m$, then by induction hypothesis we have $|s^*|_{(\mu,1)} \sim \Phi(B)^+ \sim ((\ldots (B_1 \vee B_2) \ldots \vee B_n)$ and $|s^*|_{(\mu,2)} \sim \Phi(C)^- \sim ((\ldots (C_1 \wedge C_2) \ldots \wedge C_n)$. Hence, by distributivity of $\rightarrow$ with respect to $\vee$ and $\wedge$, $|s^*|_\mu \sim ((\ldots ((\ldots (B_1 \rightarrow C_1) \wedge \ldots) \wedge (B_1 \rightarrow C_m)) \ldots \wedge ((\ldots (B_n \rightarrow C_1) \wedge \ldots) \wedge (B_n \rightarrow C_m))$. Therefore, $|s^*|_\mu \sim \Phi(A)^- = B_1 \rightarrow C_1, \ldots, B_1 \rightarrow C_m, \ldots, B_n \rightarrow C_1, \ldots, B_n \rightarrow C_m$.

$\square$

Lemma 4.7 (Validity of the Extracted Sequent)**.** Let φ be an **LK**-Proof. Then $H_{LK_A}(\varphi)$ is valid.

Proof. By lemma 4.6, $H_{LK_A}(\varphi)$ and $|\Phi_P(\Psi(\varphi))|_{[\epsilon]}$ are logically equivalent, and $|\Phi_P(\Psi(\varphi))|_{[\epsilon]}$ is valid because algorithm 4.5 constructs a proof for it. Therefore, $H_{LK_A}(\varphi)$ is also valid. $\square$

Theorem 4.8 (Soundness of Algorithm 4.4)**.** Let φ be an **LK**-Proof. Then the sequent extracted by algorithm 4.4, $H_{LK_A}(\varphi)$, is a Herbrand sequent.

Proof. Lemmas 4.5 and 4.7 show that the extracted sequent satisfies the requirements in definition 3.3.2 and thus it is a Herbrand sequent. $\square$

Definition 4.2.4 (Herbrand Sequent of an **LK**-Proof with End-sequent in Non-Prenex Form)**.** The determinism of algorithm 4.4 and theorem 4.8 together allow us to define *The Herbrand Sequent of an **LK**-Proof φ with End-Sequent in Non-Prenex Form, $H_P(\varphi)$,* as the very result of the algorithm, $H_{LK_A}(\varphi)$.

Example 4.7 (Extraction of Herbrand Sequent via Transformation from **LK** to **LK$_A$**)**.** *Let φ be the **LK**-Proof below. It is slightly modified with respect to the **LK**-Proof φ of example 4.2, because it has an end-sequent in non-prenex form, obtained from the end-sequent in example 4.2 by the additional rule $(\wedge : l)$.*

$$
\cfrac{
  \cfrac{
    P(0) \vdash P(0) \qquad
    \cfrac{
      \cfrac{
        P(s(0)) \vdash P(s(0)) \qquad
        \cfrac{
          \cfrac{
            \cfrac{P(s^2(0)) \vdash P(s^2(0)) \qquad P(s^3(0)) \vdash P(s^3(0))}{P(s^2(0)), P(s^2(0)) \rightarrow P(s^3(0)) \vdash P(s^3(0))} \rightarrow : l
          }{P(s^2(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \forall : l^{[1]}
        }{P(s(0)), P(s(0)) \rightarrow P(s^2(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \rightarrow : l
      }{
        \cfrac{P(s(0)), (\forall x)(P(x) \rightarrow P(s(x))), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))}{P(s(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \; c : l
      } \forall : l^{[2]}
  }{P(0), P(0) \rightarrow P(s(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \rightarrow : l
  }{
    \cfrac{
      \cfrac{P(0), (\forall x)(P(x) \rightarrow P(s(x))), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))}{P(0), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} \; c : l
    }{}
  } \forall : l
}{P(0) \wedge (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))} (\wedge : l)
$$

*Since its end-sequent is not in prenex-form, Mid-Sequent Reduction cannot be used to extract its Herbrand sequent. As an alternative way to extract its Herbrand sequent, we can transform φ into its **LK$_A$**-Proof $\Psi(\varphi)$, as shown below.*

*First we have to omit quantifier rules from φ, thus obtaining the following **LK**-Proof with false contractions.*

$$
\cfrac{
  P(0) \vdash P(0) \qquad
  \cfrac{
    P(s(0)) \vdash P(s(0)) \qquad
    \cfrac{
      P(s^2(0)) \vdash P(s^2(0)) \qquad P(s^3(0)) \vdash P(s^3(0))
    }{
      P(s^2(0)), P(s^2(0)) \rightarrow P(s^3(0)) \vdash P(s^3(0))
    } \; \rightarrow : l
  }{
    \cfrac{
      \cfrac{
        P(s(0)), P(s(0)) \rightarrow P(s^2(0)), P(s^2(0)) \rightarrow P(s^3(0)) \vdash P(s^3(0))
      }{
        P(s(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))
      } \; c : l
    }{}
  } \; \rightarrow : l
}{
  \cfrac{
    \cfrac{
      P(0), P(0) \rightarrow P(s(0)), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))
    }{
      \cfrac{
        P(0), (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))
      }{
        P(0) \wedge (\forall x)(P(x) \rightarrow P(s(x))) \vdash P(s^3(0))
      } \; (\wedge : l)
    } \; c : l
  }{}
} \; \rightarrow : l
$$

Then we replace. starting from above and going downwards, the 2 false contractions, $(c : l)$-rules, by $(\langle\langle : l)$-rules. The resulting $\Psi(\varphi)$ is:

$$
\cfrac{
  P(0) \vdash P(0) \qquad
  \cfrac{
    P(s(0)) \vdash P(s(0)) \qquad
    \cfrac{
      P(s^2(0)) \vdash P(s^2(0)) \qquad P(s^3(0)) \vdash P(s^3(0))
    }{
      P(s^2(0)), P(s^2(0)) \rightarrow P(s^3(0)) \vdash P(s^3(0))
    } \; \rightarrow : l
  }{
    \cfrac{
      P(s(0)), P(s(0)) \rightarrow P(s^2(0)), P(s^2(0)) \rightarrow P(s^3(0)) \vdash P(s^3(0))
    }{
      P(s(0)), \langle P(s(0)) \rightarrow P(s^2(0)), P(s^2(0)) \rightarrow P(s^3(0)) \rangle \vdash P(s^3(0))
    } \; \langle\rangle : l
  } \; \rightarrow : l
}{
  \cfrac{
    P(0), P(0) \rightarrow P(s(0)), \langle P(s(0)) \rightarrow P(s^2(0)), P(s^2(0)) \rightarrow P(s^3(0)) \rangle \vdash P(s^3(0))
  }{
    \cfrac{
      P(0), \langle P(0) \rightarrow P(s(0)), \langle P(s(0)) \rightarrow P(s^2(0)), P(s^2(0)) \rightarrow P(s^3(0)) \rangle\rangle \vdash P(s^3(0))
    }{
      P(0) \wedge \langle P(0) \rightarrow P(s(0)), \langle P(s(0)) \rightarrow P(s^2(0)), P(s^2(0)) \rightarrow P(s^3(0)) \rangle\rangle \vdash P(s^3(0))
    } \; (\wedge : l)
  } \; \langle\rangle : l
} \; \rightarrow : l
$$

We can then extract the Herbrand sequent of $\Psi(\varphi)$ by:

$$
\begin{aligned}
H(\Psi(\varphi)) \;=\;& \Phi(|\Psi(\varphi)|_{[\epsilon]}) \\[2mm]
=\;& \Phi\left(P(0) \wedge \left\langle \left\langle \begin{array}{c} P(0) \rightarrow P(s(0)), \\ P(s(0)) \rightarrow P(s^2(0)), \\ P(s^2(0)) \rightarrow P(s^3(0)) \end{array} \right\rangle \right\rangle \vdash P(s^3(0)) \right) \\[2mm]
=\;& \Phi\left(P(0) \wedge \left\langle \left\langle \begin{array}{c} P(0) \rightarrow P(s(0)), \\ P(s(0)) \rightarrow P(s^2(0)), \\ P(s^2(0)) \rightarrow P(s^3(0)) \end{array} \right\rangle \right\rangle \right) \vdash P(s^3(0)) \\[2mm]
=\;& \left(\Phi\left(P(0) \wedge \left\langle \begin{array}{c} P(0) \wedge P(0) \rightarrow P(s(0)), \\ P(s(0)) \rightarrow P(s^2(0)), \\ P(s^2(0)) \rightarrow P(s^3(0)) \end{array} \right\rangle \right) \right) \vdash P(s^3(0)) \\[2mm]
=\;& \left(\begin{array}{c} P(0) \wedge P(0) \rightarrow P(s(0)), \\ P(0) \wedge P(s(0)) \rightarrow P(s^2(0)), \\ P(0) \wedge P(s^2(0)) \rightarrow P(s^3(0)) \end{array} \right) \vdash P(s^3(0))
\end{aligned}
$$

4.2.2 Extraction via Collection of Sub-Formula Instances

The algorithm described in the previous subsection is able to extract a Herbrand sequent for a non-prenex proof, but it has the same disadvantage that the mid-sequent reduction algorithm had for the prenex case: it needs to transform the whole proof. In this subsection we describe an algorithm that extracts a Herbrand sequent for the non-prenex case without transforming the proof. This algorithm, which we call *Herbrand Sequent Extraction by Sub-Formula Instance Collection*, generalizes algorithm 4.2 using some ideas of algorithm 4.4.

We notice that algorithm 4.2 collects all the instances of all quantified occurrences of the end-sequent and then it constructs a Herbrand sequent by removing all those quantified occurrences of the end-sequent and inserting all the collected instances. However, in the non-prenex case, one difficulty arises: we need to substitute collected instances not for occurrences in the end-sequent, but for specific sub-formulas in the end-sequent. This requires two modifications:

1. we need to keep track of which instances belong to which sub-formula and collect them separately. We do this by introducing labels in the **LK**-Proof and the notion of rule-reference to a sub-formula.

2. since a quantified sub-formula in position λ may have more than one instance, as a result of possible contractions on its ancestors, we may need to substitute all these instances in the same place, in the same position λ. We solve this by grouping all these instances in a temporary array-formula. Later we expand the whole formula that contains the array-formula into a sequence of standard First-Order-Logic formulas, using the already defined function Φ to eliminate the temporary arrays.

Although the essence of the generalization, as explained above, is relatively simple, the formal specification of the algorithm requires many extensions of the definitions of positions and substitutions to deal with the temporary array-formulas. Among these extensions, the most important is the concept of *unifying substitution*. When dealing with **LK**-Proofs containing quantifiers nested in the scope of other quantifiers, it may happen that substituting formulas do not unify with the sub-formulas they were supposed to substitute, and these substitutions should be avoided, because performing them would imply the deletion or corruption of instantiations due to another contracted copy of the outer quantified formula or to another branch of the tree. Unifying substitutions must distinguish different contracted copies of a sub-formula, and in order for this to be possible, we also define markers for sub-formulas. Example 4.17 shows an **LK**-Proof with nested quantifiers in which unifying substitutions play a big role.

As for the previous algorithm, *Extraction via Transformation to* **LK$_{\mathbf{A}}$**, in this algorithm it is also assumed that the proofs are skolemized.

Definition 4.2.5 (Position-labeled Sequent)**.** A *position-labeled sequent s* is a sequent in which each occurrence is labeled with a position (μ, λ), where μ is a position of a occurrence in s and λ is a position of a sub-formula in this occurrence.

Definition 4.2.6 (Position-labeled **LK**-Proof)**.** A *position-labeled **LK**-Proof* is defined recursively as an **LK**-Proof in which:

- Each occurrence in position μ_i in the end-sequent is labeled with (μ_i, ε).

- For each rule, the context occurrences in the premise have the same labels as their corresponding occurrences in the conclusion.

- For each rule, let (μ_m, λ_m) be the label of its main occurrence. Then its auxiliary occurrences are labeled with $(\mu_m, \lambda_m.j)$, where j is the position of main occurrence's sub-formula corresponding to the auxiliary occurrence.

- Cut-formulas (and their ancestors) are not labeled.

Example 4.8 (Position-labeled **LK**-Proof)**.** *The **LK**-Proof φ below is shown with position labels in the upper-right corner of each occurrence:*

$$
\cfrac{
  \cfrac{
    P(0)^{(A.1,1)} \vdash P(0)^{(A.1,211)}
    \qquad
    P(s(0))^{(A.1,212)}, (\forall x)(P(x) \to P(s(x)))^{(A.1,2)} \vdash P(s^3(0))^{(C.1,\varepsilon)} \quad [\varphi_1]
  }{
    \cfrac{
      \cfrac{
        P(0)^{(A.1,1)}, P(0) \to P(s(0))^{(A.1,21)}, (\forall x)(P(x) \to P(s(x)))^{(A.1,2)} \vdash P(s^3(0))^{(C.1,\varepsilon)}
      }{
        \cfrac{
          P(0)^{(A.1,1)}, (\forall x)(P(x) \to P(s(x)))^{(A.1,2)}, (\forall x)(P(x) \to P(s(x)))^{(A.1,2)} \vdash P(s^3(0))^{(C.1,\varepsilon)}
        }{
          P(0)^{(A.1,1)}, (\forall x)(P(x) \to P(s(x)))^{(A.1,2)} \vdash P(s^3(0))^{(C.1,\varepsilon)}
        } \; c:l
      } \; \forall:l
  }
  }{
    (P(0) \wedge (\forall x)(P(x) \to P(s(x))))^{(A.1,\varepsilon)} \vdash P(s^3(0))^{(C.1,\varepsilon)}
  } \; \wedge:l
} \; \to:l
$$

where φ_1 is:

$$
\cfrac{
  \cfrac{
    P(s(0))^{(A.1,212)} \vdash P(s(0))^{(A.1,211)} \qquad [\varphi_2]
  }{
    \cfrac{
      P(s(0))^{(A.1,212)}, P(s(0)) \to P(s^2(0))^{(A.1,21)}, (\forall x)(P(x) \to P(s(x)))^{(A.1,2)} \vdash P(s^3(0))^{(C.1,\varepsilon)}
    }{
      P(s(0))^{(A.1,212)}, (\forall x)(P(x) \to P(s(x)))^{(A.1,2)}, (\forall x)(P(x) \to P(s(x)))^{(A.1,2)} \vdash P(s^3(0))^{(C.1,\varepsilon)}
    } \; \forall:l^{[2]}
  }
}{
  P(s(0))^{(A.1,212)}, (\forall x)(P(x) \to P(s(x)))^{(A.1,2)} \vdash P(s^3(0))^{(C.1,\varepsilon)})
} \; c:l
$$

where φ_2 is:

$$
\cfrac{
  P(s^2(0))^{(A.1,212)} \vdash P(s^2(0))^{(A.1,211)} \qquad P(s^3(0))^{(A.1,212)} \vdash P(s^3(0))^{(C.1,\varepsilon)}
}{
  \cfrac{
    P(s^2(0))^{(A.1,212)}, P(s^2(0)) \to P(s^3(0))^{(A.1,21)} \vdash P(s^3(0))^{(C.1,\varepsilon)}
  }{
    P(s^2(0))^{(A.1,212)}, (\forall x)(P(x) \to P(s(x)))^{(A.1,2)} \vdash P(s^3(0))^{(C.1,\varepsilon)}
  } \; \forall:l^{[1]}
} \; \to:l
$$

Definition 4.2.7 (Marking Atomic Sub-Formulas: Contractions and Depth)**.** The *marker* of an atomic sub-formula F occurring in position $[\nu, \mu, \lambda]$ of a labeled **LK**-Proof φ is denoted by $\mathcal{M}([\nu, \mu, \lambda])$, and it is a string belonging to $\mathbb{N}^*$. All sub-formulas from φ are marked, starting from the bottom and going upwards, according to the inductive rules below:

- Base Case 1: If F is an atomic sub-formula in the end-sequent ($\nu = \varepsilon$), then $\mathcal{M}([\nu, \mu, \lambda]) = 1$.

- Base Case 2: If F is an atomic cut-formula, then $\mathcal{M}([\nu, \mu, \lambda]) = 1$.

- Inductive Case: Let ρ be a rule in φ and let $\overline{m}$ be the marker of a sub-formula F in its conclusion sequent. If ρ is a contraction rule and F is its main occurrence, then we mark all ancestors F_j of F with $\overline{m}j$. Otherwise, we mark the ancestors of F with $\overline{m}1$.

Example 4.9 (Marking Atomic Sub-Formulas). *In the proof φ below, we show the marker of each atomic sub-formula as strings subscripted to the predicate symbol.*

$$\frac{\dfrac{P_{1111111}(a, b) \vdash P_{1111111}(a, b) \qquad P_{1211111}(a, c) \vdash P_{1111111}(a, c)}{P_{111111}(a, b), P_{121111}(a, c) \vdash P_{111111}(a, b) \wedge P_{111111}(a, c)} \wedge : l}{\dfrac{\dfrac{P_{11111}(a, b), (\forall y)(P_{12111}(a, y)) \vdash P_{11111}(a, b) \wedge P_{11111}(a, c)}{(\forall y)(P_{1111}(a, y)), (\forall y)(P_{1211}(a, y)) \vdash P_{1111}(a, b) \wedge P_{1111}(a, c)} \forall : l}{\dfrac{(\forall y)(P_{111}(x, y)), (\forall x)(\forall y)(P_{121}(x, y)) \vdash P_{111}(a, b) \wedge P_{111}(a, c)}{\dfrac{(\forall x)(\forall y)(P_{11}(x, y)), (\forall x)(\forall y)(P_{12}(x, y)) \vdash P_{11}(a, b) \wedge P_{11}(a, c)}{(\forall x)(\forall y)(P_1(x, y)) \vdash P_1(a, b) \wedge P_1(a, c)} c : l} \forall : l} \forall : l}$$

The markers provide us with two kinds of information:

- *The length of the marker informs the depth in the proof-tree in which the sub-formula occurs.*

- *Whenever a contraction occurs in a given depth of the tree, we have numbers different from 1 in the corresponding position of the marker. These numbers keep track of different copies of the atomic sub-formula that were contracted.*

Definition 4.2.8 (Marking Atomic Sub-Formulas: Contractions and Contraction-Depth). The *contraction-marker* of an atomic sub-formula F occurring in position $[\nu, \mu, \lambda]$ of a labeled **LK**-Proof φ is denoted by $\mathcal{M}_{\mathcal{C}}([\nu, \mu, \lambda])$, and it is a string belonging to $\mathbb{N}^*$. All sub-formulas from φ are marked, starting from the bottom and going upwards, according to the inductive rules below:

- Base Case 1: If F is an atomic sub-formula in the end-sequent ($\nu = \varepsilon$), then $\mathcal{M}_{\mathcal{C}}([\nu, \mu, \lambda]) = \varepsilon$.

- Base Case 2: If F is an atomic cut-formula, then $\mathcal{M}_{\mathcal{C}}([\nu, \mu, \lambda]) = \varepsilon$.

- Inductive Case: Let ρ be a rule in φ and let $\overline{m}$ be the marker of a sub-formula F in its conclusion sequent. If ρ is a contraction rule and F is its main occurrence, then we mark all ancestors F_j of F with $\overline{m}j$. Otherwise, we mark the ancestors of F with $\overline{m}$.

The length of the contraction-marker of a sub-formula is called *contraction-depth* of the sub-formula.

Example 4.10 (Marking Atomic Sub-Formulas: Contractions and Contraction-Depth). *Considering example 4.9, we notice that the markers serve the purpose of distinguishing contracted copies, but since we are usually not interested in the depth of sub-formulas, those sequences of 1 generated whenever a sub-formula in the conclusion has only one copy in the premises is useless and only prevents the easy reading of the interesting information contained in the markers. Marking with Contraction-markers, however, adds new numbers to the string only when a sub-formula has more than one copy in the premises, making it easier to read and distinguish contracted copies. The* **LK**-*Proof below is the same shown in example 4.9, but with contraction-markers instead of simple markers:*

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{P_1(a,b) \vdash P(a,b) \qquad P_2(a,c) \vdash P(a,c)}{P_1(a,b), P_2(a,c) \vdash P(a,b) \land P(a,c)} \land : l}{P_1(a,b), (\forall y)(P_2(a,y)) \vdash P(a,b) \land P(a,c)} \forall : l}{(\forall y)(P_1(a,y)), (\forall y)(P_2(a,y)) \vdash P(a,b) \land P(a,c)} \forall : l}{(\forall y)(P_1(x,y)), (\forall x)(\forall y)(P_2(x,y)) \vdash P(a,b) \land P(a,c)} \forall : l}{(\forall x)(\forall y)(P_1(x,y)), (\forall x)(\forall y)(P_2(x,y)) \vdash P(a,b) \land P(a,c)} \forall : l}{(\forall x)(\forall y)(P(x,y)) \vdash P(a,b) \land P(a,c)} c : l$$

Contraction-markers provide us with two kinds of information:

- *The length of the marker, the contraction-depth, informs the number of contractions (implicit or explicit) suffered by the sub-formula in the path from the sub-formula to its terminal formula (a formula in the end-sequent or a cut-formula).*

- *Whenever a contraction occurs the copies being contracted have different numbers in the last position of their contraction-markers. This allow us to distinguish them.*

Definition 4.2.9 (Rule-Reference to Sub-formula). We say that a rule refers to a sub-formula in position λ of a formula occurrence in position μ of the end-sequent if and only if its main occurrence is labeled with (μ, λ).

Definition 4.2.10 (Set of Rules Referring to a Sub-Formula). $Q_{(\mu,\lambda)}(\varphi)$ is the set of all rules of the **LK**-Proof φ referring to the sub-formula in position λ of the formula occurrence in position μ in the end-sequent.

Definition 4.2.11 (Sets of Used Auxiliary Occurrences). Let R be a set of rules. With $\mathcal{A}(R)$ we denote the set of used auxiliary occurrences in the rules in R.

Definition 4.2.12 (Set to Array Formation). Let M be a set of occurrences. With $\mathcal{S}(M)$ we denote the array-formula formed with the occurrences in M.

Definition 4.2.13 (Array-Formula Referring to a Sub-Formula). Let λ be a position in an occurrence in position μ of the end-sequent of a proof φ.

Then we define the array-formula referring to the sub-formula in position λ of the occurrence indexed with μ in the end-sequent:

$$Q^A_{(\mu,\lambda)}(\varphi) \doteq \mathcal{S}(\mathcal{A}(Q_{(\mu,\lambda)}(\varphi))))$$

Definition 4.2.14 (Dequantification of a Sequent)**.** Let s be a sequent with all variables standardized apart. Then $\mathcal{D}(s)$ denotes the sequent obtained from s by removing all quantifiers and substituting all the variables that were bound by these quantifiers by distinct eigen-constants.

Definition 4.2.15 (Positions of an Array-Formula)**.** Let A be an array-formula, then its set of *positions* $pos(A)$ is the smallest set of strings belonging to $\{1,2\}^*$ and such that $pos(A)$ satisfies:

1. $\varepsilon \in pos(A)$

2. If A is of the form $\neg B$, then $\{1.\lambda | \lambda \in pos(B)\} \subset pos(A)$.

3. If A is of the form $(B \wedge C)$, then $(\{1.\lambda_1 | \lambda_1 \in pos(B)\} \cup \{2.\lambda_2 | \lambda_2 \in pos(C)\}) \subset pos(A)$.

4. If A is of the form $(B \vee C)$, then $(\{1.\lambda_1 | \lambda_1 \in pos(B)\} \cup \{2.\lambda_2 | \lambda_2 \in pos(C)\}) \subset pos(A)$.

5. If A is of the form $(B \to C)$, then $(\{1.\lambda_1 | \lambda_1 \in pos(B)\} \cup \{2.\lambda_2 | \lambda_2 \in pos(C)\}) \subset pos(A)$.

6. If A is of the form $(\forall x)B$, then $\{1.\lambda | \lambda \in pos(B)\} \subset pos(A)$.

7. If A is of the form $(\exists x)B$, then $\{1.\lambda | \lambda \in pos(B)\} \subset pos(A)$.

8. If A is of the form $\langle A_1, A_2, \ldots, A_n \rangle$, then $\{1.\lambda | \lambda \in pos(A_j) \text{ for some } j \in \{1,\ldots,n\}\} \subset pos(A)$

Definition 4.2.16 (Multi-sets of Sub-formulas occurring in given Positions of an Array-Formula)**.** Let λ be a position of the array-formula A. Then the multi-set of *sub-formulas* occurring in position λ is denoted $|A|_\lambda$. Formally, we may define the computation of $|A|_\lambda$ inductively:

1. $|A|_\varepsilon \doteq \{A\}$

2. If A is of the form $\neg B$ and $\lambda = 1.\lambda'$ with $\lambda' \in pos(B)$, then $|A|_\lambda \doteq |B|_{\lambda'}$.

3. If A is of the form $B \circ C$ (for $\circ \in \{\wedge, \vee, \to\}$) and $\lambda = 1.\lambda'$ with $\lambda' \in pos(B)$, then $|A|_\lambda \doteq |B|_{\lambda'}$.

4. If A is of the form $B \circ C$ (for $\circ \in \{\wedge, \vee, \to\}$) and $\lambda = 2.\lambda'$ with $\lambda' \in pos(C)$, then $|A|_\lambda \doteq |C|_{\lambda'}$.

5. If A is of the form $(Qx)B$ (for $Q \in \{\forall, \exists\}$) and $\lambda = 1.\lambda'$ with $\lambda' \in pos(B)$, then $|A|_\lambda \doteq |B|_{\lambda'}$.

6. If A is of the form $\langle A_1, A_2, \ldots, A_n \rangle$, and $\lambda = 1.\lambda'$, then $|A|_\lambda \doteq \bigcup_{\lambda' \in pos(A_j), j \in \{1, \ldots, n\}} \{|A_j|_{\lambda'}\}$.

Example 4.11 (Multi-sets of Sub-Formulas occurring in given Positions of an Array-Formula). *Let* $A \doteq \langle P(a) \wedge (\forall y)(Q(a, y) \rightarrow Q(a, s(y))), P(b) \wedge (\forall y)(Q(b, y) \rightarrow Q(b, s(y)))\rangle$ *be an array-formula. Then:*

$$|A_\varepsilon = \{\langle P(a) \wedge (\forall y)(Q(a, y) \rightarrow Q(a, s(y))), P(b) \wedge (\forall y)(Q(b, y) \rightarrow Q(b, s(y)))\rangle\}$$

$$|A|_{11} = \{P(a), P(b)\}$$

$$|A|_{12} = \{(\forall y)(Q(a, y) \rightarrow Q(a, s(y))), (\forall y)(Q(b, y) \rightarrow Q(b, s(y)))\}$$

Definition 4.2.17 (Sub-Formula Substitutions in Array-Formulas). Let λ be a position of the array-formula A such that $|A|_\lambda = \{B_1, B_2, \ldots, B_n\}$. The array-formula obtained by substituting the B_j in position λ in A by C is denoted $A_\lambda[C]$ or $A\sigma^{fo}$, where $\sigma^{fo} \doteq \{\lambda/C\}$.

Example 4.12 (Sub-Formula Substitutions in Array-Formulas). *Let* $A = \langle P(a) \wedge (\forall y)(Q(a, y)), P(b) \wedge (\forall y)(Q(b, y))\rangle$ *be an array-formula. Then:*

$$A\{2/Q(a, 0)\} = \langle P(a) \wedge Q(a, 0), P(b) \wedge Q(a, 0)\rangle$$

Definition 4.2.18 (Marker-Unification). Two formulas A and B marker-unify, $Unifiable_M(A, B)$, if and only if they unify and the contraction-markers of the atomic sub-formulas of A (or B) are prefix substrings of the contraction-markers of the corresponding sub-formulas of B (or A). Marker-unification is thus unification aware of contraction-markers.

Example 4.13 (Marker-unification). *Marker-unification is standard unification extended to formulas with contraction-markers, as exemplified below:*

- $Unifiable_M(P(a), P(x))$.

- $Unifiable_M(P_{12}(a), P(x))$.

- $Unifiable_M(P_{12}(a), P_1(x))$.

- $\neg Unifiable_M(P_{12}(a), P_2(x))$.

- $Unifiable_M(Q_{12}(a) \wedge P_{12}(a), Q_1(a) \wedge P_1(x))$.

- $\neg Unifiable_M(Q_{12}(a) \wedge P_{12}(a), Q_2(a) \wedge P_2(x))$.

Definition 4.2.19 (Unifying Substitutions in Array-Formulas). Let λ be a position of the array-formula A such that $|A|_\lambda = \{B_1, B_2, \ldots, B_n\}$. Let $C \doteq \langle C_1, \ldots, C_m \rangle$. Let $A_Q \doteq \{j | B_j(\text{ is of the form } (Qx)B'_j\}$ with $Q \in \{\forall, \exists\}$. For $j \in A_Q$, let $U_j \doteq \{C_k | C_k \in |C|_1 \text{ and } Unifiable_M(C_k, B'_j)\}$. Let $A'_Q \doteq \{j | U_j \neq \emptyset\}$. then, by $A\overline{\sigma^{fo}}$, where $\overline{\sigma^{fo}} \doteq \overline{\{\lambda/C\}}$, we denote the array-formula obtained by substituting in A those B_j ($j \in A'_Q$) in position λ by the array formula $\mathcal{S}(U_j)$.

Furthermore, $\overline{\sigma^{fs}}$ and $\overline{\sigma^{fp}}$ are defined analogously.

Example 4.14 (Unifying Substitutions in Array-Formulas). *Let $A = \langle P(a) \wedge (\forall y)(Q(a,y)), P(b) \wedge (\forall y)(Q(b,y)) \rangle$ be an array-formula. Then:*

$$A\overline{\{2/Q(a,0)\}} = \langle P(a) \wedge Q(a,0), P(b) \wedge (\forall y)(Q(b,y)) \rangle$$

because $Q(a,0)$ unifies with $Q(a,y)$, but not with $Q(b,y)$. The difference between usual substitutions and unifying substitutions can be seen by comparing this example with example 4.12

Example 4.15 (Unifying Substitutions with Contraction-markers). *Let $A \doteq (\forall x)(\forall y)P(x,y)$. Then:*

$$A\overline{\{\varepsilon/\langle (\forall y)P_1(a,y), (\forall y)P_2(a,y) \rangle\}} = \langle (\forall y)P_1(a,y), (\forall y)P_2(a,y) \rangle$$

$$A\overline{\{\varepsilon/\langle (\forall y)P_1(a,y), (\forall y)P_2(a,y) \rangle\}} \circ \overline{\{1/\langle P_1(a,b), P_2(a,c) \rangle\}} = \langle \langle P_1(a,b) \rangle, \langle P_2(a,c) \rangle \rangle$$

$$A\overline{\{\varepsilon/\langle (\forall y)P_1(a,y), (\forall y)P_2(a,y) \rangle\}} \circ \overline{\{1/\langle P_1(a,b) \rangle\}} \circ \overline{\{1/\langle P_2(a,c) \rangle\}} = \langle \langle P_1(a,b) \rangle, \langle P_2(a,c) \rangle \rangle$$

For the sake of contrast, without contraction-markers we would have:

$$A\overline{\{\varepsilon/\langle (\forall y)P(a,y), (\forall y)P(a,y) \rangle\}} = \langle (\forall y)P(a,y), (\forall y)P(a,y) \rangle$$

$$A\overline{\{\varepsilon/\langle (\forall y)P(a,y), (\forall y)P(a,y) \rangle\}} \circ \overline{\{1/\langle P(a,b), P(a,c) \rangle\}} = \langle \langle P(a,b), P(a,c) \rangle, \langle P(a,b), P(a,c) \rangle \rangle$$

$$A\overline{\{\varepsilon/\langle (\forall y)P(a,y), (\forall y)P(a,y) \rangle\}} \circ \overline{\{1/\langle P(a,b) \rangle\}} \circ \overline{\{1/\langle P(a,c) \rangle\}} = \langle \langle P(a,b) \rangle, \langle P(a,b) \rangle \rangle$$

By comparing the last substitutions of the cases with and without contraction-markers, we can see their importance. Without contraction-markers, the instantiation of $(\forall y)P(a,y)$ to $P(a,c)$ was lost, because $P(a,y)$ was firstly unified with $P(a,b)$ substituted by it.

Algorithm 4.6 (Extraction of Herbrand Sequent without Proof-Transformation for Proofs with Non-prenex End-sequents). Let φ be a Position-labeled **LK**-Proof, and let s be its end-sequent. Let:

$$S_{QSF}(s) \doteq \{(\mu_i, \lambda_i) | \lambda_i \text{ is a position such that } |s|_{(\mu_i, \lambda_i)} \text{ is a sub-formula starting with a quantifier}\}$$

For each position $(\mu_i, \lambda_i) \in S_{QSF}(s)$, let its corresponding substitution be:

$$\sigma^{fs}_{(\mu_i,\lambda_i)} \doteq \{(\mu_i,\lambda_i)/Q^A_{(\mu_i,\lambda_i)}(\varphi)\}$$

Then we compute the following sequent from φ:

$$H_A(\varphi) \doteq set(P(\Phi(s\overline{(\sigma^{fs}_{(\mu_0,\lambda_0)} \circ \ldots \circ \sigma^{fs}_{(\mu_n,\lambda_n)})})))$$

where: $depth(\lambda_i) < depth(\lambda_j)$, if $i < j$ and $\mu_i = \mu_j$ [1]

Algorithm 4.2.2: $H_A(\varphi)$

$s \leftarrow$ end-sequent of φ
$S_{QSF} \leftarrow$ positions of quantified sub-formulas of s
for each $(\mu_i, \lambda_i) \in S_{QSF}$
$\quad$ **do** $\begin{cases} Q_{(\mu,\lambda)} \leftarrow \text{REFERENCINGRULES}(\varphi, (\mu, \lambda)) \\ \mathcal{A}_{(\mu,\lambda)} \leftarrow \text{USEDAUXILIARYOCCURRENCES}(Q_{(\mu,\lambda)}) \\ Q^A_{(\mu,\lambda)} \leftarrow \text{ARRAYFORMATION}(\mathcal{A}_{(\mu,\lambda)}) \\ \sigma^{fs}_{(\mu_i,\lambda_i)} \leftarrow \{(\mu_i,\lambda_i)/Q^A_{(\mu_i,\lambda_i)}\} \end{cases}$
$s_H \leftarrow s$
$\text{SORTBYDEPTH}(S_{QSF})$
for each $(\mu_i, \lambda_i) \in S_{QSF}$
$\quad$ **do** $\left\{ s_H \leftarrow s_H \overline{\sigma^{fs}_{(\mu_i,\lambda_i)}} \right.$
$s_H \leftarrow \Phi(s_H)$
$s_H \leftarrow \text{DEQUANTIFICATION}(s_H)$
$s_H \leftarrow \text{SETNORMALIZATION}(s_H)$
return (s_H)

Lemma 4.8 (Extracted Sequent Well-Instantiated). Let φ be an **LK**-Proof and let its end-sequent be $|\varphi|_{[\epsilon]} = A_1, \ldots, A_n \vdash B_1, \ldots, B_m$.
$A^0_1, \ldots, A^0_n, B^0_1, \ldots, B^0_m$ are its formulas without quantifiers. Then the sequent extracted by algorithm 4.6 has the form:

$$H_A(\varphi) = A^1_1, \ldots, A^{i_1}_1, \ldots A^1_n, \ldots, A^{i_n}_n \vdash B^1_1, \ldots, B^{p_1}_1, \ldots, B^1_m, \ldots, B^{p_m}_m$$

where: $A^1_j, \ldots, A^{i_j}_j$ are quantifier-free substitution instances of A^0_j and $B^1_k, \ldots, B^{p_k}_k$ are quantifier-free substitution instances of B^0_k.

Proof. By definition of Φ ([1]). $\qquad\qquad\square$

---

[1]This condition on the order of the composition of substitutions guarantees that variables bound by outer quantifiers are substituted before variables bound to nested inner quantifiers. This is important, because otherwise the substitution of an outer bounded variable would erase the substitution of an inner bounded variable.

Lemma 4.9 (Validity of the Extracted Sequent)**.** Let φ be an **LK**-Proof. Then $H_A(\varphi)$ is valid.

Proof. Here we informally sketch a proof by induction.

In order to prove that $H_A(\varphi)$ is valid, we show a proof transformation that can be used to transform φ into an **LK**-Proof φ^* of $H_A(\varphi)$ [2]. This transformation progressively eliminates quantifier-rules from φ and adjusts the sequents so that the resulting proof always remains a valid **LK**-Proof.

- *Base Case; Quantifiers with depth 0*:

  For a quantifier with depth 0 in an occurrence in position μ of the end-sequent, we transform the proof by deleting all quantifier-rules and all contraction-rules referring to (μ, ε) and by performing a unifying substitution of the sub-formula in position (μ, ε) by

  $$\Phi(Q^A_{(\mu,\varepsilon)}(\varphi))$$

- *Induction Case; Quantifiers with depth $n + 1$*:

  Let φ' be the proof after all quantifiers up to depth n have been eliminated. And we are now interested in eliminating a quantifier with depth $n + 1$ in a sub-formula in position (μ, λ) in the end-sequent.

  Let $\nu_1, \ldots, \nu_n$ be positions in φ' such that $\varphi'_{[\nu_i]}$ has an ancestor occurrence of $|\varphi'|_{[\varepsilon,\mu]}$ such that $|\varphi'|_{[\nu,\mu_i,\varepsilon]} = |\varphi'|_{[\varepsilon,\mu,\lambda]}$.

  We transform $|\varphi'|_{\nu_i}$ by deleting all quantifier-rules and contraction-rules referring to (μ_i, ε) and by performing a unifying substitution [3] of the sub-formula in position (μ_i, ε) by

  $$\Phi(Q^A_{(\mu,\lambda)}(\varphi))$$

  Additionally, if the formula occurrence at position $[\nu_i, \mu_i, \epsilon]$ is an ancestor of some sub-formula F_s in position λ_s of an occurrence F, then we perform a unifying substitution of F by

  $$\Phi(F\sigma^{fo}_s)$$

  where

  $$\sigma^{fo}_s \doteq \{\lambda_s/Q^A_{(\mu,\lambda)}(\varphi)\}$$

---

[2] Alternatively, a similar, easier but more indirect, proof could be done by transforming φ into a valid **LK$_A$**-Proof.

[3] We do a unifying substitution instead of just a normal substitution, because in general the sub-formula in position $(\mu_i, 1)$ may have had some of its variables instantiated by outer quantifiers and therefore it may not be unifiable with the formulas in $Q^A_{(\mu,\lambda)}(\varphi)$. In the case they do not unify, we do not want to perform the substitution, since this would affect the instantiation that was done by outer quantifiers.

Let φ_1 be the result of applying the transformations above for all quantifiers in the end-sequent of φ. If some of the quantified sub-formulas of φ were not used (in other words, if they don't have an ancestor in an axiom), then φ_1 will still contain some quantifiers in the end-sequent. We therefore transform φ_1 into φ_2 by dequantification (in other words, we substitute the remaining quantified variables in sub-formulas of the end-sequent and in their ancestor sub-formulas by eigen-constants).

Since $Q^A_{(\mu,\lambda)}(\varphi)$ is an array-formula which may have more than one formula possibly coming from different proof branches, φ_2 may have excessive formulas in some branches. The instantiation brings to one branch instantiated formulas that are actually necessary only in another branch. These excessive formulas must be generated and removed by inserting extra weakenings and by doubling some propositional rules in φ_2, in order to produce a well-formed **LK**-Proof φ^*.

By the way we constructed φ^*, eliminating the quantifiers from φ by substituting them by the corresponding array formula with their instances, it is clear that the end-sequent of φ^* is $H_A(\varphi)$.

Example 4.18 shows this process of transforming an **LK**-Proof φ into an **LK**-Proof φ^* for its extracted Herbrand sequent. $\square$

Example 4.16 (Extraction of Herbrand Sequent without Proof Transformation). *Here we consider the same proof φ of example 4.7, but now with labels, which however are shown below only for the end-sequent s:*

$$
\cfrac{
  \cfrac{
    \cfrac{
      \cfrac{
        \cfrac{
          \cfrac{P(s^2(0)) \vdash P(s^2(0)) \qquad P(s^3(0)) \vdash P(s^3(0))}{P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0))} \to: l
        }{P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \forall : l^{[1]}
      }{
        \cfrac{P(s(0)) \vdash P(s(0)) \qquad \ldots}{P(s(0)), P(s(0)) \to P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \to: l
      }
    }{\ldots}
  }{\ldots}
}{(P(0) \wedge (\forall x)(P(x) \to P(s(x))))^{(A.1,\varepsilon)} \vdash P(s^3(0))^{(C.1,\varepsilon)}} \wedge : l
$$

Firstly we compute the set of positions corresponding to sub-formulas beginning with quantifiers in the quantifiers:

$$S_{QSF}(\varphi) = \{(A.1, 2)\}$$

Then we compute the sets of quantifier rules referring to the sub-formulas beginning with quantifiers. In our case, there is only one such a set, namely $Q_{(A.1,2)}(\varphi)$, with the following rules:

$$
\cfrac{P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \vdash P(s^3(0))}{P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \forall : l^{[1]}
$$

$$\frac{P(s(0)), P(s(0)) \to P(s^2(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))}{P(s(0)), (\forall x)(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \ \forall : l^{[2]}$$

$$\frac{P(0), P(0) \to P(s(0)), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))}{P(0), (\forall x)(P(x) \to P(s(x))), (\forall x)(P(x) \to P(s(x))) \vdash P(s^3(0))} \ \forall : l$$

Its corresponding sets of used auxiliary occurrences are:

$$\mathcal{A}(Q_{(A.1,2)}(\varphi)) = \left\{ \begin{array}{l} P(0) \to P(s(0)), \\ P(s(0)) \to P(s^2(0)), \\ P(s^2(0)) \to P(s^3(0)) \end{array} \right\}$$

The array-formula formed from these used occurrences is:

$$Q^A_{(A.1,2)}(\varphi) = \langle P(0) \to P(s(0)), P(s(0)) \to P(s^2(0)), P(s^2(0)) \to P(s^3(0)) \rangle$$

We can then construct the corresponding substitution of sub-formulas in sequents:

$$\begin{aligned} \sigma^{fs}_{(A.1,2)} &= \{(A.1,2)/Q^A_{(A.1,2)}(\varphi)\} \\ &= \left\{ (A.1,2) / \left\langle \begin{array}{l} P(0) \to P(s(0)), \\ P(s(0)) \to P(s^2(0)), \\ P(s^2(0)) \to P(s^3(0)) \end{array} \right\rangle \right\} \end{aligned}$$

Finally, we can extract the Herbrand sequent of the proof:

$$\begin{aligned} H_A(\varphi)) &= set(\mathcal{D}(\Phi(s\overline{\sigma^{fs}_{(A.1,2)}}))) \\ &= set\left(P\left(\Phi\left(P(0) \wedge \left\langle \begin{array}{l} P(0) \to P(s(0)), \\ P(s(0)) \to P(s^2(0)), \\ P(s^2(0)) \to P(s^3(0)) \end{array} \right\rangle \vdash P(s^3(0)) \right) \right) \right) \\ &= set\left(\mathcal{D}\left(\left(\begin{array}{l} P(0) \wedge P(0) \to P(s(0)), \\ P(0) \wedge P(s(0)) \to P(s^2(0)), \\ P(0) \wedge P(s^2(0)) \to P(s^3(0)) \end{array} \vdash P(s^3(0)) \right) \right) \right) \\ &= set\left(\left(\begin{array}{l} P(0) \wedge P(0) \to P(s(0)), \\ P(0) \wedge P(s(0)) \to P(s^2(0)), \\ P(0) \wedge P(s^2(0)) \to P(s^3(0)) \end{array} \vdash P(s^3(0)) \right) \right) \\ &= \left(\begin{array}{l} P(0) \wedge P(0) \to P(s(0)), \\ P(0) \wedge P(s(0)) \to P(s^2(0)), \\ P(0) \wedge P(s^2(0)) \to P(s^3(0)) \end{array} \vdash P(s^3(0)) \right) \end{aligned}$$

Example 4.17 (Extraction of Herbrand Sequent without Proof Transformation and with Nested Quantifiers)**.** *Let φ be the proof below:*

$$
\cfrac{
\cfrac{\varphi_1 \qquad \varphi_2}{
\left(
\begin{array}{c}
Q(a,0), \\
(\forall x)(P(x) \wedge (\forall y)(Q(x,y) \rightarrow Q(x,s(y)))), \\
(\forall x)(P(x) \wedge (\forall y)(Q(x,y) \rightarrow Q(x,s(y))))
\end{array}
\right) \vdash Q(a,s^2(0)) \wedge (P(a) \wedge P(b))
} \; \wedge : r
}{
\left(
\begin{array}{c}
, Q(a,0), \\
(\forall x)(P(x) \wedge (\forall y)(Q(x,y) \rightarrow Q(x,s(y))))
\end{array}
\right) \vdash Q(a,s^2(0)) \wedge (P(a) \wedge P(b))
} \; c : l
$$

where φ_1 is:

$$
\cfrac{
\cfrac{
\cfrac{
Q(a,0) \vdash Q(a,0) \qquad
\cfrac{
\cfrac{
\cfrac{Q(a,s(0)) \vdash Q(a,s(0)) \qquad Q(a,s^2(0)) \vdash Q(a,s^2(0))}{Q(a,s(0)), Q(a,s(0)) \rightarrow Q(a,s^2(0)) \vdash Q(a,s^2(0))} \; \rightarrow : l
}{Q(a,s(0)), (\forall y)(Q(a,y) \rightarrow Q(a,s(y))) \vdash Q(a,s^2(0))} \; \forall : l
}{\, }
}{Q(a,0), Q(a,0) \rightarrow Q(a,s(0)), (\forall y)(Q(a,y) \rightarrow Q(a,s(y))) \vdash Q(a,s^2(0))} \; \rightarrow : l
}{
\cfrac{
\cfrac{
\cfrac{
\cfrac{Q(a,0), (\forall y)(Q(a,y) \rightarrow Q(a,s(y))), (\forall y)(Q(a,y) \rightarrow Q(a,s(y))) \vdash Q(a,s^2(0))}{Q(a,0), (\forall y)(Q(a,y) \rightarrow Q(a,s(y))) \vdash Q(a,s^2(0))} \; c : l
}{Q(a,0), P(a), (\forall y)(Q(a,y) \rightarrow Q(a,s(y))) \vdash Q(a,s^2(0))} \; w : l
}{Q(a,0), P(a) \wedge (\forall y)(Q(a,y) \rightarrow Q(a,s(y))) \vdash Q(a,s^2(0))} \; \wedge : l
}{Q(a,0), (\forall x)(P(x) \wedge (\forall y)(Q(x,y) \rightarrow Q(x,s(y)))) \vdash Q(a,s^2(0))} \; \forall : l
} \; \forall : l
$$

and φ_2 is:

$$
\cfrac{
\cfrac{[\varphi_{21}] \qquad [\varphi_{22}]}{
\left(
\begin{array}{c}
(\forall x)(P(x) \wedge (\forall y)(Q(x,y) \rightarrow Q(x,s(y)))), \\
(\forall x)(P(x) \wedge (\forall y)(Q(x,y) \rightarrow Q(x,s(y))))
\end{array}
\right) \vdash P(a) \wedge P(b)
}
}{(\forall x)(P(x) \wedge (\forall y)(Q(x,y) \rightarrow Q(x,s(y)))) \vdash P(a) \wedge P(b)} \; c : l
$$

with φ_{21}:

$$
\cfrac{
\cfrac{
\cfrac{
\cfrac{P(a) \vdash P(a)}{(P(a), (\forall y)(Q(a,y) \rightarrow Q(a,s(y)))) \vdash P(a)} \; w : l
}{(P(a) \wedge (\forall y)(Q(a,y) \rightarrow Q(a,s(y)))) \vdash P(a)} \; \wedge : l
}{(\forall x)(P(x) \wedge (\forall y)(Q(x,y) \rightarrow Q(x,s(y)))) \vdash P(a)} \; \forall : l
}{\,}
$$

and φ_{22}:

$$
\cfrac{
\cfrac{
\cfrac{
\cfrac{P(b) \vdash P(b)}{(P(b), (\forall y)(Q(b,y) \rightarrow Q(b,s(y)))) \vdash P(b)} \; w : l
}{(P(b) \wedge (\forall y)(Q(b,y) \rightarrow Q(b,s(y)))) \vdash P(b)} \; \wedge : l
}{(\forall x)(P(x) \wedge (\forall y)(Q(x,y) \rightarrow Q(x,s(y)))) \vdash P(b)} \; \forall : l
}{\,}
$$

Firstly we compute the set of positions corresponding to sub-formulas beginning with quantifiers in the end-sequent:

$$ S_{QSF}(\varphi) = \{(A.2, \varepsilon), (A.2, 12)\} $$

Then we compute the sets of quantifier rules referring to the sub-formulas beginning with quantifiers. In the case of this example, there are two such sets (because the cardinality of S_{QSF} is 2). The first set, $Q_{(A.2,\varepsilon)}(\varphi)$, has the following rules:

$$\frac{Q(a,0), P(a) \wedge (\forall y)(Q(a,y) \rightarrow Q(a,s(y))) \vdash Q(a,s^2(0))}{Q(a,0), (\forall x)(P(x) \wedge (\forall y)(Q(x,y) \rightarrow Q(x,s(y)))) \vdash Q(a,s^2(0))} \forall : l$$

$$\frac{(P(a) \wedge (\forall y)(Q(a,y) \rightarrow Q(a,s(y)))) \vdash P(a)}{(\forall x)(P(x) \wedge (\forall y)(Q(x,y) \rightarrow Q(x,s(y)))) \vdash P(a)} \forall : l$$

$$\frac{(P(b) \wedge (\forall y)(Q(b,y) \rightarrow Q(b,s(y)))) \vdash P(b)}{(\forall x)(P(x) \wedge (\forall y)(Q(x,y) \rightarrow Q(x,s(y)))) \vdash P(a)} \forall : l$$

The second set, $Q_{(A.2,12)}$, has the following rules:

$$\frac{Q(a,s(0)), Q(a,s(0)) \rightarrow Q(a,s^2(0)) \vdash Q(a,s^2(0))}{Q(a,s(0)), (\forall y)(Q(a,y) \rightarrow Q(a,s(y))) \vdash Q(a,s^2(0))} \forall : l$$

$$\frac{\left(\begin{array}{c} Q(a,0), \\ Q(a,0) \rightarrow Q(a,s(0)), \\ (\forall y)(Q(a,y) \rightarrow Q(a,s(y))) \end{array} \right) \vdash Q(a,s^2(0))}{\left(\begin{array}{c} Q(a,0), \\ (\forall y)(Q(a,y) \rightarrow Q(a,s(y))), \\ (\forall y)(Q(a,y) \rightarrow Q(a,s(y))) \end{array} \right) \vdash Q(a,s^2(0))} \forall : l$$

The sets of used auxiliary occurrences are:

$$\mathcal{A}(Q_{(A.2,\varepsilon)}(\varphi)) = \left\{ \begin{array}{c} P(a) \wedge (\forall y)(Q(a,y) \rightarrow Q(a,s(y))), \\ P(b) \wedge (\forall y)(Q(b,y) \rightarrow Q(b,s(y)))) \end{array} \right\}$$

and:

$$\mathcal{A}(Q_{(A.2,12)}(\varphi)) = \{Q(a,s(0)) \rightarrow Q(a,s^2(0)), Q(a,0) \rightarrow Q(a,s(0))\}$$

The array-formulas formed from these used occurrences are:

$$Q^A_{(A.2,\varepsilon)}(\varphi) = \left\langle \begin{array}{c} P(a) \wedge (\forall y)(Q(a,y) \rightarrow Q(a,s(y))), \\ P(b) \wedge (\forall y)(Q(b,y) \rightarrow Q(b,s(y)))) \end{array} \right\rangle$$

$$Q^A_{(A.2,12)}(\varphi) = \langle Q(a,s(0)) \rightarrow Q(a,s^2(0)), Q(a,0) \rightarrow Q(a,s(0)) \rangle$$

We can then construct the corresponding 2 substitutions of sub-formulas in sequents:

$$\begin{aligned} \sigma^{fs}_{(A.2,\varepsilon)} &= \{(A.2,\varepsilon)/Q^A_{(A.2,\varepsilon)}(\varphi)\} \\ &= \left\{ (A.2,\varepsilon)/ \left\langle \begin{array}{c} P(a) \wedge (\forall y)(Q(a,y) \rightarrow Q(a,s(y))), \\ P(b) \wedge (\forall y)(Q(b,y) \rightarrow Q(b,s(y)))) \end{array} \right\rangle \right\} \end{aligned}$$

$$\begin{aligned} \sigma^{fs}_{(A.2,12)} &= \{(A.2,12)/Q^A_{(A.2,12)}(\varphi)\} \\ &= \{(A.2,12)/\langle Q(a,s(0)) \rightarrow Q(a,s^2(0)), Q(a,0) \rightarrow Q(a,s(0)) \rangle\} \end{aligned}$$

Finally, we can extract the Herbrand sequent of the proof (explanations on each of the derivation steps are given subsequently):

$$
\begin{aligned}
H_A(\varphi) \quad =^1 \quad & set(P(\Phi(s(\overline{\sigma^{fs}_{(A.2,\varepsilon)} \circ \sigma^{fs}_{(A.2,12)}})))) \\[4pt]
=^2 \quad & set(\mathcal{D}(\Phi((Q(a,0), \\
& \quad (\forall x)(P(x) \wedge (\forall y)(Q(x,y) \to Q(x,s(y)))) \\
& \quad \vdash Q(a,s^2(0)) \wedge (P(a) \wedge P(b)))\overline{(\sigma^{fs}_{(A.2,\varepsilon)} \circ \sigma^{fs}_{(A.2,12)})})))) \\[4pt]
=^3 \quad & set(\mathcal{D}(\Phi(((Q(a,0), \\
& \quad (\forall x)(P(x) \wedge (\forall y)(Q(x,y) \to Q(x,s(y)))) \\
& \quad \vdash Q(a,s^2(0)) \wedge (P(a) \wedge P(b)))\overline{\sigma^{fs}_{(A.2,\varepsilon)}})\overline{\sigma^{fs}_{(A.2,12)}}))) \\[4pt]
=^4 \quad & set(\mathcal{D}(\Phi((Q(a,0), \\
& \quad \langle P(a) \wedge (\forall y)(Q(a,y) \to Q(a,s(y))), \\
& \quad P(b) \wedge (\forall y)(Q(b,y) \to Q(b,s(y))))\rangle \\
& \quad \vdash Q(a,s^2(0)) \wedge (P(a) \wedge P(b)))\overline{\sigma^{fs}_{(A.2,12)}}))) \\[4pt]
=^5 \quad & set(\mathcal{D}(\Phi((Q(a,0), \\
& \quad \langle P(a) \wedge \langle Q(a,s(0)) \to Q(a,s^2(0)), \\
& \quad Q(a,0) \to Q(a,s(0))\rangle, \\
& \quad P(b) \wedge \langle(\forall y)(Q(b,y) \to Q(b,s(y)))\rangle\rangle \\
& \quad \vdash Q(a,s^2(0)) \wedge (P(a) \wedge P(b))))))) \\[4pt]
=^6 \quad & set(\mathcal{D}(\Phi((Q(a,0), \\
& \quad P(a) \wedge \langle Q(a,s(0)) \to Q(a,s^2(0)), \\
& \quad Q(a,0) \to Q(a,s(0))\rangle, \\
& \quad P(b) \wedge \langle(\forall y)(Q(b,y) \to Q(b,s(y)))\rangle \\
& \quad \vdash Q(a,s^2(0)) \wedge (P(a) \wedge P(b))))))) \\[4pt]
=^7 \quad & set(\mathcal{D}(\Phi((Q(a,0), \\
& \quad P(a) \wedge (Q(a,s(0)) \to Q(a,s^2(0))), \\
& \quad P(a) \wedge (Q(a,0) \to Q(a,s(0))), \\
& \quad P(b) \wedge \langle(\forall y)(Q(b,y) \to Q(b,s(y)))\rangle \\
& \quad \vdash Q(a,s^2(0)) \wedge (P(a) \wedge P(b))))))) \\[4pt]
=^8 \quad & set(\mathcal{D}((Q(a,0), \\
& \quad P(a) \wedge (Q(a,s(0)) \to Q(a,s^2(0))), \\
& \quad P(a) \wedge (Q(a,0) \to Q(a,s(0))), \\
& \quad P(b) \wedge (\forall y)(Q(b,y) \to Q(b,s(y))) \\
& \quad \vdash Q(a,s^2(0)) \wedge (P(a) \wedge P(b)))))
\end{aligned}
$$

$$
\begin{aligned}
=^9 \quad & set((Q(a,0), \\
& \quad P(a) \wedge (Q(a,s(0)) \to Q(a,s^2(0))), \\
& \quad P(a) \wedge (Q(a,0) \to Q(a,s(0))), \\
& \quad P(b) \wedge ((Q(b,c^*) \to Q(b,s(c^*)) \\
& \quad \vdash Q(a,s^2(0)) \wedge (P(a) \wedge P(b)))) \\
=^{10} \quad & Q(a,0), \\
& \quad P(a) \wedge (Q(a,s(0)) \to Q(a,s^2(0))), \\
& \quad P(a) \wedge (Q(a,0) \to Q(a,s(0))), \\
& \quad P(b) \wedge (Q(b,c^*) \to Q(b,s(c^*))) \\
& \quad \vdash Q(a,s^2(0)) \wedge (P(a) \wedge P(b))
\end{aligned}
$$

Explanations for each derivation step:

1. *Expansion of the definition of $H_A(\varphi)$.*

2. *Expansion of the end-sequent s.*

3. *Expansion of the definition of the composition of substitutions.*

4. *Application of the unifying substitution $\overline{\sigma^{fs}_{(A.2,\varepsilon)}}$.*
   Since $(P(x) \wedge (\forall y)(Q(x,y) \to Q(x,s(y))))$ unifies with the formulas in $Q^A_{(A.2,\varepsilon)}(\varphi)$ ($(P(a) \wedge (\forall y)(Q(a,y) \to Q(a,s(y))))$ and $(P(b) \wedge (\forall y)(Q(b,y) \to Q(b,s(y)))))$, the occurrence $(\forall x)(P(x) \wedge (\forall y)(Q(x,y) \to Q(x,s(y))))$ is substituted by $Q^A_{(A.2,\varepsilon)}(\varphi)$.

5. *Application of the unifying substitution $\overline{\sigma^{fs}_{(A.2,12)}}$. Since $(Q(a,y) \to Q(a,s(y)))$ unifies with with the formulas in $Q^A_{(A.2,12)}(\varphi)$ ($Q(a,s(0)) \to Q(a,s^2(0))$ and $Q(a,0) \to Q(a,s(0))$), $(\forall y)(Q(a,y) \to Q(a,s(y)))$ is substituted by $Q^A_{(A.2,12)}(\varphi)$. On the other hand $(Q(b,y) \to Q(b,s(y)))$, which is also in position $(A.2,12)$, does not unify with the formulas in $Q^A_{(A.2,12)}(\varphi)$, because the term b clashes with the term a. Therefore $(\forall y)(Q(b,y) \to Q(b,s(y)))$ remains unchanged (not substituted). This exemplifies the importance of performing a unifying substitution instead of just a normal substitution. Different branches of the tree required the quantified x to be instantiated to b and to a. Then only the branch of the tree responsible for the instantiation of x to a required the quantified y to be instantiated, according to the formulas in $Q^A_{(A.2,12)}(\varphi)$. If we had performed a normal substitution, we would have undesirably erased the instantiation of x to b due to the other branch of the tree. Unification is thus used to take care of branching and contractions in the **LK**-Proof.*

6. *First iteration in the expansion of array-formulas, by way of Φ.*

7. *Second iteration in the expansion of array-formulas, by way of Φ.*

8. *Third iteration in the expansion of array-formulas, by way of Φ.*

9. *Dequantification of the sequent. Since $(\forall y)(Q(b, y) \rightarrow Q(b, s(y)))$ is not used (it appeared only by weakenings in φ), it was not substituted by any unifying substitution. Hence an undesired quantifier remained in the sequent. The dequantification simply removes the quantifier and substitutes y by an eigen-constant c∗, without affecting the validity of the sequent, since the fact that it appeared by weakening means that this sub-formula is actually irrelevant for the proof. We could have simply removed the sub-formula completely, but the definition of Herbrand sequent requires instantiations of the complete formulas.*

10. *Set-normalization of the sequent.*

Example 4.18 (Constructing a Proof for $H_A(\varphi)$). *Let φ be the **LK**-Proof below:*

$$
\cfrac{
  \cfrac{[\varphi^0] \qquad [\varphi^1]}{(P(0) \wedge (\forall y)(Q(y))) \wedge (\forall x)(P(x)) \vdash P(0) \wedge P(1)} \qquad [\varphi^2]
}{
  (P(0) \wedge (\forall y)(Q(y))) \wedge (\forall x)(P(x)) \vdash (P(0) \wedge P(1)) \wedge P(2)
}
$$

where φ^0 is:

$$
\cfrac{
  \cfrac{
    \cfrac{
      \cfrac{
        \cfrac{P(0) \vdash P(0)}{P(0), (\forall x)(P(x)) \vdash P(0)} \; w : l
      }{P(0), (\forall y)(Q(y)), (\forall x)(P(x)) \vdash P(0)} \; w : l
    }{(P(0) \wedge (\forall y)(Q(y))), (\forall x)(P(x)) \vdash P(0)} \; \wedge : l
  }{(P(0) \wedge (\forall y)(Q(y))) \wedge (\forall x)(P(x)) \vdash P(0)} \; \wedge : l
}{}
$$

and φ^1 is:

$$
\cfrac{
  \cfrac{
    \cfrac{
      \cfrac{
        \cfrac{
          \cfrac{P(1) \vdash P(1)}{(\forall x)P(x) \vdash P(1)} \; \forall : l
        }{P(0), (\forall x)P(x) \vdash P(1)} \; w : l
      }{P(0), (\forall y)Q(y), (\forall x)P(x) \vdash P(1)} \; w : l
    }{(P(0) \wedge (\forall y)Q(y)), (\forall x)P(x) \vdash P(1)} \; \wedge : l
  }{(P(0) \wedge (\forall y)Q(y)) \wedge (\forall x)P(x) \vdash P(1)} \; \wedge : l
}{}
$$

and φ^2 is:

$$\frac{\dfrac{\dfrac{\dfrac{\dfrac{P(2) \vdash P(2)}{(\forall x)P(x) \vdash P(2)} \; \forall : l}{P(0), (\forall x)P(x) \vdash P(2)} \; w : l}{P(0), (\forall y)Q(y), (\forall x)P(x) \vdash P(2)} \; w : l}{(P(0) \wedge (\forall y)Q(y)), (\forall x)P(x) \vdash P(2)} \; \wedge : l}{(P(0) \wedge (\forall y)Q(y)) \wedge (\forall x)P(x) \vdash P(2)} \; \wedge : l$$

Its Herbrand sequent, extracted by algorithm 4.6 is:

$$H_A(\varphi) = (P(0) \wedge (Q(c_1^*))) \wedge P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash (P(0) \wedge P(1)) \wedge P(2)$$

*Now we transform φ into an **LK**-Proof φ^* of this Herbrand Sequent. Firstly, we perform the unifying substitutions in the inductive way explained in Lemma 4.9, and we obtain the pseudo-proof φ_1 below, where false inference rules are marked with $*$:*

$$\frac{\begin{array}{cc} [\varphi_1^0] & [\varphi_1^1] \end{array}}{\dfrac{(P(0) \wedge (\forall y)(Q(y))) \wedge (\forall x)(P(x)) \vdash P(0) \wedge P(1) \qquad [\varphi_1^2]}{(P(0) \wedge (\forall y)(Q(y))) \wedge P(1), (P(0) \wedge (\forall y)(Q(y))) \wedge P(2) \vdash (P(0) \wedge P(1)) \wedge P(2)}}$$

where φ_1^0 is:

$$\frac{\dfrac{\dfrac{\dfrac{\dfrac{P(0) \vdash P(0)}{P(0), P(1), P(2) \vdash P(0)} \; w : l}{P(0), (\forall y)(Q(y)), P(1), P(2) \vdash P(0} \; w : l}{(P(0) \wedge (\forall y)(Q(y))), P(1), P(2) \vdash P(0)} \; \wedge : l}{(P(0) \wedge (\forall y)(Q(y))) \wedge P(1), (P(0) \wedge (\forall y)(Q(y))) \wedge P(2) \vdash P(0)} \; \wedge : l^*$$

and φ_1^1 is:

$$\frac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{P(1) \vdash P(1)}{P(1), P(2) \vdash P(1)} \; \forall : l}{P(0), P(1), P(2) \vdash P(1)} \; w : l}{P(0), (\forall y)(Q(y)), P(1), P(2) \vdash P(1)} \; w : l}{(P(0) \wedge (\forall y)(Q(y))), P(1), P(2) \vdash P(1)} \; \wedge : l}{(P(0) \wedge (\forall y)(Q(y))) \wedge P(1), (P(0) \wedge (\forall y)(Q(y))) \wedge P(2) \vdash P(1)} \; \wedge : l^*$$

and φ_1^2 is:

$$\frac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{P(2) \vdash P(2)}{P(1), P(2) \vdash P(2)} \; \forall : l}{P(0), P(1), P(2) \vdash P(2)} \; w : l}{P(0), (\forall y)(Q(y)), P(1), P(2) \vdash P(2)} \; w : l}{(P(0) \wedge (\forall y)(Q(y))), P(1), P(2) \vdash P(2)} \; \wedge : l}{(P(0) \wedge (\forall y)(Q(y))) \wedge P(1), (P(0) \wedge (\forall y)(Q(y))) \wedge P(2) \vdash P(2)} \; \wedge : l^*$$

Then we perform dequantification and we obtain φ_2 below:

$$\cfrac{\begin{array}{cc}[\varphi_2^0] & [\varphi_2^1]\end{array}}{(P(0) \wedge Q(c_1^*)) \wedge P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash P(0) \wedge P(1)} \quad [\varphi_2^2]$$
$$\cfrac{}{(P(0) \wedge Q(c_1^*)) \wedge P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash (P(0) \wedge P(1)) \wedge P(2)}$$

where φ_2^0 is:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{P(0) \vdash P(0)}{P(0), P(1), P(2) \vdash P(0)} \; w:l}{P(0), Q(c_1^*), P(1), P(2) \vdash P(0)} \; w:l}{(P(0) \wedge Q(c_1^*)), P(1), P(2) \vdash P(0)} \; \wedge:l}{(P(0) \wedge Q(c_1^*)) \wedge P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash P(0)} \; \wedge:l^*}$$

and φ_2^1 is:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{P(1) \vdash P(1)}{P(1), P(2) \vdash P(1)} \; \forall:l}{P(0), P(1), P(2) \vdash P(1)} \; w:l}{P(0), Q(c_1^*), P(1), P(2) \vdash P(1)} \; w:l}{(P(0) \wedge Q(c_1^*)), P(1), P(2) \vdash P(1)} \; \wedge:l}{(P(0) \wedge Q(c_1^*)) \wedge P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash P(1)} \; \wedge:l^*}$$

and φ_2^2 is:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{P(2) \vdash P(2)}{P(1), P(2) \vdash P(2)} \; \forall:l}{P(0), P(1), P(2) \vdash P(2)} \; w:l}{P(0), Q(c_2^*), P(1), P(2) \vdash P(2)} \; w:l}{(P(0) \wedge Q(c_2^*)), P(1), P(2) \vdash P(2)} \; \wedge:l}{(P(0) \wedge Q(c_1^*)) \wedge P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash P(2)} \; \wedge:l^*}$$

*Finally, we insert the missing weakenings and propositional-rules, in order to obtain the desired **LK**-Proof φ^*:*

$$\cfrac{\begin{array}{cc}[\varphi_0^*] & [\varphi_1^*]\end{array}}{(P(0) \wedge Q(c_1^*)) \wedge P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash P(0) \wedge P(1)} \quad [\varphi_2^*]$$
$$\cfrac{}{(P(0) \wedge Q(c_1^*)) \wedge P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash (P(0) \wedge P(1)) \wedge P(2)}$$

where φ_0^* is:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{P(0) \vdash P(0)}{P(0), P(1) \vdash P(0)} \; w:l}{P(0), P(1), P(2) \vdash P(0)} \; w:l}{P(0), Q(c_1^*), P(1), P(2) \vdash P(0)} \; w:l}{(P(0) \wedge Q(c_1^*)), P(1), P(2) \vdash P(0)} \; \wedge:l}{(P(0) \wedge Q(c_1^*)) \wedge P(1), P(2) \vdash P(0)} \; \wedge:l}{(P(0) \wedge Q(c_1^*)) \wedge P(1), (P(0) \wedge Q(c_2^*)), P(2) \vdash P(0)} \; w:l}{(P(0) \wedge Q(c_1^*)) \wedge P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash P(0)} \; \wedge:l$$

and φ_1^* is:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{P(1) \vdash P(1)}{P(1), P(2) \vdash P(1)} \; w:l}{P(0), P(1), P(2) \vdash P(1)} \; w:l}{P(0), Q(c_1^*), P(1), P(2) \vdash P(1)} \; w:l}{(P(0) \wedge Q(c_1^*)), P(1), P(2) \vdash P(1)} \; \wedge:l}{(P(0) \wedge Q(c_1^*)) \wedge P(1), P(2) \vdash P(1)} \; \wedge:l}{(P(0) \wedge Q(c_1^*)) \wedge P(1), P(0) \wedge Q(c_2^*), P(2) \vdash P(1)} \; w:l}{(P(0) \wedge Q(c_1^*)) \wedge P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash P(1)} \; \wedge:l$$

and φ_2^* is:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{P(2) \vdash P(2)}{P(1), P(2) \vdash P(2)} \; \forall:l}{P(0), P(1), P(2) \vdash P(2)} \; w:l}{P(0), Q(c_2^*), P(1), P(2) \vdash P(2)} \; w:l}{(P(0) \wedge Q(c_2^*)), P(1), P(2) \vdash P(2)} \; \wedge:l}{P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash P(2)} \; \wedge:l}{(P(0) \wedge Q(c_1^*)), P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash P(2)} \; w:l}{(P(0) \wedge Q(c_1^*)) \wedge P(1), (P(0) \wedge Q(c_2^*)) \wedge P(2) \vdash P(2)} \; \wedge:l$$

Chapter 5

Conclusion

The main contribution of this thesis is the development of an algorithm for the extraction of Herbrand sequents from formal proofs written in the Sequent Calculus **LK**. This algorithm draws ideas from two previously existing algorithms in a way that combines their strengths but avoids their weaknesses. Namely, the algorithm here developed also accepts proofs in non-prenex form, and it doesn't need to perform expensive proof transformations to obtain the desired Herbrand sequent.

The resulting Herbrand sequent acts as a summarization of the interesting content of the formal proof, thus improving its human-understandability. However, since the algorithm is restricted to the Sequent Calculus **LK** for First-Order Logic, it remains for future work to investigate how it could be adapted and extended to other calculi and other logics.

Bibliography

[1] M.Baaz; A.Leitsch. On skolemization and proof complexity. *Fundamenta Matematicae*, (20):353–379, 1994.

[2] Matthias Baaz, Stefan Hetzl, Alexander Leitsch, Clemens Richter, and Hendrik Spohr. Ceres: An Analysis of Fürstenberg's Proof of the Infinity of Primes. in preparation.

[3] Matthias Baaz, Stefan Hetzl, Alexander Leitsch, Clemens Richter, and Hendrik Spohr. Cut-Elimination: Experiments with CERES. In Franz Baader and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR) 2004*, volume 3452 of *Lecture Notes in Computer Science*, pages 481–495. Springer, 2005.

[4] Matthias Baaz, Stefan Hetzl, Alexander Leitsch, Clemens Richter, and Hendrik Spohr. Proof Transformation by CERES. In Jonathan M. Borwein and William M. Farmer, editors, *Mathematical Knowledge Management (MKM) 2006*, volume 4108 of *Lecture Notes in Artificial Intelligence*, pages 82–93. Springer, 2006.

[5] S. R. Buss. On Herbrand's theorem. *Lecture Notes in Computer Science*, 960:195, 1995.

[6] G. Gentzen. Untersuchungen über das logische schließen. In M.E.Szabo, editor, *The Collected Papers of Gerhard Gentzen*, pages 68–131. North-Holland Publishing Company, Amsterdam - London, 1969.

[7] Stefan Hetzl. The two dual parts of formal proofs. In preparation.

[8] Stefan Hetzl. Projection-based cut-elimination and normalization. Diplom, Technische Universitt Wien, Wien, March 2004.

[9] J.Herbrand. *Recherches sur la Theorie de la Demonstration*. PhD thesis, University of Paris, 1930.

[10] M. Baaz; A. Leitsch. Cut normal forms and proof complexity. *Annals of Pure and Applied Logic*, 97:127–177, 1999.

[11] G. Sutcliffe and C. Suttner. The State of CASC. *AI Communications*,
     19(1):35–48, 2006.