Herbrand-Confluence for Cut Elimination in Classical First Order Logic
Stefan Hetzl, Lutz Strassburger

To cite this version:
Stefan Hetzl, Lutz Strassburger. Herbrand-Confluence for Cut Elimination in Classical First Order Logic. CSL 2012, Sep 2012, Fontainebleau, France. 10.4230/LIPIcs.CSL.2012.320 . hal-00759228

HAL Id: hal-00759228
https://inria.hal.science/hal-00759228
Submitted on 3 Dec 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Herbrand-Confluence for Cut Elimination in
Classical First Order Logic

Stefan Hetzl¹ and Lutz Straßburger²

1 Institute of Discrete Mathematics and Geometry
Vienna University of Technology
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria
hetzl@logic.at

2 INRIA Saclay – Île-de-France
Ecole Polytechnique, LIX
Rue de Saclay, 91128 Palaiseau Cedex, France
lutz@lix.polytechnique.fr

Abstract

We consider cut-elimination in the sequent calculus for classical first-order logic. It is well known that this system, in its most general form, is neither confluent nor strongly normalizing. In this work we take a coarser (and mathematically more realistic) look at cut-free proofs. We analyze which witnesses they choose for which quantifiers, or in other words: we only consider the Herbrand-disjunction of a cut-free proof. Our main theorem is a confluence result for a natural class of proofs: all (possibly infinitely many) normal forms of the non-erasing reduction lead to the same Herbrand-disjunction.

1998 ACM Subject Classification

Keywords and phrases
proof theory, first-order logic, tree languages, term rewriting, semantics of proofs

Digital Object Identifier 10.4230/LIPIcs.CSL.2012.320

1 Introduction

The constructive content of proofs has always been a central topic of proof theory and it is also one of the most important influences that logic has on computer science. Classical logic is widely used and presents interesting challenges when it comes to understanding the constructive content of its proofs. These challenges have therefore attracted considerable attention, see, for example, [24, 11, 10], [6], [26, 27], [8], [21], or [5], for different investigations in this direction.

A well-known, but not yet well-understood, phenomenon is that a single classical proof usually allows several different constructive readings. From the point of view of applications this means that we have a choice among different programs that can be extracted. In [25] the authors show that two different extraction methods applied to the same proof produce two programs, one of polynomial and one of exponential average-case complexity. This phenomenon is further exemplified by case studies in [26, 3, 4] as well as the asymptotic results [2, 15]. The reason for this behavior is that classical “proofs often leave algorithmic detail underspecified” [1].

On the level of cut-elimination in the sequent calculus this phenomenon is reflected by the fact that the standard proof reduction without imposing any strategy is not confluent. In this
paper we consider cut-elimination in classical first-order logic and treat the question which cut-free proofs one can obtain (by the strategy-free rewriting system) from a single proof with cuts. As our aim is to compare cut-free proofs we need a notion of equivalence of proofs: clearly the syntactic equality makes more differences than those which are mathematically interesting. Being in a system with quantifiers, a natural and more realistic choice is to consider two cut-free proofs equivalent if they choose the same terms for the same quantifiers, in other words: if they have the same Herbrand-disjunction.

A cut-reduction relation will then be called Herbrand-confluent if all its normal forms have the same Herbrand-disjunction. The main result of this paper is that, for a natural class of proofs, the standard reduction without erasing of subproofs is Herbrand-confluent. This result is surprising as this reduction is neither confluent nor strongly normalizing and may produce normal forms of arbitrary size (which—as our result shows—arise only from repetitions of the same instances).

As a central proof technique we use rigid tree languages which have been introduced in [19] with applications in verification (e.g. of cryptographic protocols as in [20]) as their primary purpose. To a proof we will associate a rigid tree grammar whose language is invariant under non-erasing cut-elimination and hence equal to the only obtainable Herbrand-disjunction. This property suggests the new notion of Herbrand-content of a proof, which is defined as the language of the grammar of the proof, and which is a strong invariant. A side effect of this proof technique is a combinatorial description of how the structure of a cut-free proof is related to that of a proof with cut. Such descriptions are important theoretical results which underlie applications such as algorithmic cut-introduction as in [18].

In Section 2 we briefly review the sequent calculus and cut-elimination for classical first-order logic. In Section 3 we describe regular and rigid tree grammars which we relate to proofs in Section 4. Section 5 is devoted to proving the invariance of the Herbrand-content under duplication of subproofs, and finally, in Section 6, we collect all results together.

2 Sequent Calculus and Cut-Elimination

For the sake of simplicity, we consider only a one-sided sequent calculus and formulas in negation normal form, but the results can be proved for a two-sided sequent calculus in the same way.

Definition 1. A proof is a tree of multisets of formulas. Axioms are of the form \( \overline{A} \) for \( A \) atomic (where \( \overline{A} \) denotes the De Morgan-dual of \( A \)). The inference rules are:

\[
\Gamma, \forall x A \quad \Gamma, \exists x A \quad \Gamma, A, \overline{A} \quad \Gamma, A, A \quad \Gamma, A, _c \quad \Gamma, A, \overline{A} \quad \Gamma, \Delta, A \land B \quad \Gamma, A, B \quad \Gamma, A, \overline{A}, \Delta \quad \Gamma, A, \Delta
\]

where \( \alpha \) is called eigenvariable and does not appear in \( \Gamma, \forall x A \) and \( t \) does not contain a bound variable. We use the notation \( [x/\alpha] \) for the substitution that replaces \( x \) by the eigenvariable \( \alpha \). Similarly, \( [x/t] \) is the substitution that replaces \( x \) with \( t \).

The explicitly mentioned formula in a conclusion of an inference rule, like \( A \lor B \) for \( \lor \) is called main formula. Analogously, the explicitly mentioned formulas in the premises of an inference rule, like \( A \) and \( B \) for \( \lor \), are called auxiliary formulas. In the context of a concrete derivation we speak about main and auxiliary occurrences of inferences.

Definition 2. A proof is called regular if different \( \forall \)-inferences have different eigenvariables.

We use the following convention: We use lowercase Greek letters \( \alpha, \beta, \gamma, \delta, \ldots \) for eigenvariables in proofs, and \( \pi, \psi, \ldots \) for proofs. For a proof \( \pi \) we write \( EV(\pi) \) for the set of
Axiom reduction:

\[
\frac{\psi}{\Gamma, A \quad \exists x \ A} \quad \Gamma, A\]

Quantifier reduction:

\[
\frac{\psi_1 \Delta, \exists x \ A}{\psi_2 \Delta, \forall x \ A, \Gamma} \quad \frac{\psi_3 \Delta, A[x \alpha]}{\psi_4 \Delta, A[x \alpha], \forall} \quad \frac{\psi_5 \Delta, A[x \alpha], \Gamma}{\psi_6 \Delta, A[x \alpha], \Gamma} \quad \text{cut}
\]

Propositional reduction:

\[
\frac{\psi_1 \Delta, B}{\psi_2 \Delta, A \land B} \quad \frac{\psi_3 \Delta, A \lor B, \Pi}{\psi_4 \Delta, A \lor B, \Pi} \quad \frac{\psi_5 \Delta, B}{\psi_6 \Delta, A \land B} \quad \frac{\psi_7 \Delta, A \lor B, \Pi}{\psi_8 \Delta, A \lor B, \Pi} \quad \text{cut}
\]

Contraction reduction:

\[
\frac{\psi_1 \Gamma, A, A}{\psi_2 \Gamma, A} \quad \frac{\psi_3 \Gamma, A}{\psi_4 \Gamma, A, \Delta} \quad \frac{\psi_5 \Gamma, A, \Delta}{\psi_6 \Gamma, A, \Delta} \quad \frac{\psi_7 \Gamma, A, \Delta}{\psi_8 \Gamma, A, \Delta} \quad \text{cut}
\]

Weakening reduction:

\[
\frac{\psi_1 \Gamma, A}{\psi_2 \Gamma, A, \Delta} \quad \frac{\psi_3 \Gamma, A}{\psi_4 \Gamma, A, \Delta} \quad \text{cut}
\]

Unary inference permutation:

\[
\frac{\psi_1 \Gamma, A}{\psi_2 \Gamma, A, \Delta} \quad \frac{\psi_3 \Gamma, A}{\psi_4 \Gamma, A, \Delta} \quad \text{cut}
\]

Binary inference permutation:

\[
\frac{\psi_1 \Gamma, A}{\psi_2 \Gamma, A, \Delta} \quad \frac{\psi_3 \Gamma, A}{\psi_4 \Gamma, A, \Delta} \quad \text{cut}
\]

\[\boxed{\text{Figure 1}}\] Cut-reduction steps.

eigenvariables of $\forall$-inferences of $\pi$. Furthermore, we write $|\pi|$ for the number of inferences in $\pi$. Our results do not depend on technical differences in the definition of the calculus (which in classical logic are inessential) such as the choice between multiplicative and additive rules and the differences in the cut-reduction induced by these choices. However, for the sake of precision, let us formally define the cut-reduction we use in this paper.
Definition 3. Cut-reduction is defined on regular proofs and consists of the proof rewrite steps shown in Figure 1 (as well as all their symmetric variants), where in the contraction reduction step $\rho' = [\alpha' \backslash \alpha]_\vdash_{\psi_2}$ and $\rho'' = [\alpha' \backslash \alpha']_\vdash_{\psi_2}$ are substitutions replacing each eigenvariable occurrence $\alpha$ in $\psi_2$ by fresh copies, i.e., $\alpha'$ and $\alpha''$ are fresh for the whole proof. We write $\rightsquigarrow$ for the compatible (w.r.t. the inference rules), reflexive and transitive closure of $\rightsquigarrow$.

The above system for cut-reduction consists of purely local, minimal steps and therefore allows the simulation of many other reduction relations. We chose to work in this system in order to obtain invariance results of maximal strength. Among the systems that can be simulated literally are for example all color annotations of [11] in the multiplicative version of LK defined there. The real strength of the results in this paper lies however in the general applicability of the used proof techniques: the extraction of a grammar from a proof (that is described in the next sections) is possible in all versions of sequent calculus for classical logic and in principle also in other systems like natural deduction.

3 Regular and Rigid Tree Grammars

Formal language theory constitutes one of the main areas of theoretical computer science. Traditionally, a formal language is defined to be a set of strings but this notion can be generalized in a straightforward way to considering a language to be a set of first-order terms. Such tree languages possess a rich theory and many applications, see e.g. [13], [9]. In this section we introduce notions and results from the theory of tree languages that we will use for our proof-theoretic purposes.

A ranked alphabet $\Sigma$ is a finite set of symbols which have an associated arity (their rank). We write $\mathcal{T}_\Sigma$ to denote the set of all finite trees (or terms) over $\Sigma$, and we write $\mathcal{F}_\Sigma(X)$ to denote the set of all trees over $\Sigma$ and a set $X$ of variables (seen as symbols of arity 0). We also use the notion of position in a tree, which is a list of natural numbers. We write $\varepsilon$ for the empty list (the root position), and we write $p.q$ for the concatenation of lists $p$ and $q$. We write $p \leq q$ if $p$ is a prefix of $q$ and $p < q$ if $p$ is a proper prefix of $q$. Clearly, $\leq$ is a partial order and $<$ is its strict part. We write $\text{Pos}(t)$ to denote the set of all position in a term $t \in \mathcal{T}_\Sigma(X)$.

Definition 4. A regular tree grammar is a tuple $G = \langle N, \Sigma, \theta, P \rangle$, where $N$ is a finite set of non-terminal symbols, and $\Sigma$ is a ranked alphabet, such that $N \cap \Sigma = \emptyset$, $\theta$ is the start symbol with $\theta \in N$, and $P$ is a finite set of production rules of the form $\beta \rightarrow t$ with $\beta \in N$ and $t \in \mathcal{T}_\Sigma(N)$.

The derivation relation $\rightsquigarrow_G$ of a regular tree grammar $G = \langle N, \Sigma, \theta, P \rangle$ is defined as follows. We have $s \rightsquigarrow_G r$ if there is a production rule $\beta \rightarrow t$ in $P$ and a position $p \in \text{Pos}(s)$, such that $s|_p = \beta$ and $r$ is obtained from $s$ by replacing $\beta$ at $p$ by $t$. The language of $G$ is then defined as $L(G) = \{ t \in \mathcal{T}_\Sigma \mid \theta \rightsquigarrow_G^* t \}$, where $\rightsquigarrow_G^*$ is the transitive, reflexive closure of $\rightsquigarrow_G$. A derivation $\mathcal{D}$ of a term $t \in L(G)$ is a sequence $t_0 \rightsquigarrow_G t_1 \rightsquigarrow_G \ldots \rightsquigarrow_G t_n$ with $t_0 = \theta$ and $t_n = t$. Note that a term $t$ might have different derivations in $G$.

In [19] the class of rigid tree languages has been introduced with applications in verification (e.g. of cryptographic protocols as in [20]) as primary motivation. It will turn out that this class is appropriate for describing cut-elimination in classical first-order logic. In contrast to [19] we do not use automata but grammars—their equivalence is shown in [17].

Definition 5. A rigid tree grammar is a tuple $\langle N, N_R, \Sigma, \theta, P \rangle$, where $\langle N, \Sigma, \theta, P \rangle$, is a
regular tree grammar and \( N_R \subseteq N \) is the set of rigid non-terminals. We speak of a totally rigid tree grammar if \( N_R = N \). In this case we will just write \((N_R, \Sigma, \theta, P)\).

A derivation \( \theta = t_0 \to_G t_1 \to_G \ldots \to_G t_n = t \) of a rigid tree grammar \( G = (N, N_R, \Sigma, \theta, P) \) is a derivation in the underlying regular tree grammar satisfying the additional rigidity condition: If there are \( i, j < n \), a non-terminal \( \beta \in N_R \), and positions \( p \) and \( q \) such that \( t_i[p] = \beta \) and \( t_j[q] = \beta \) then \( t_i[p] = t_j[q] \). The language \( L(G) \) of the rigid tree grammar \( G \) is the set of all terms \( t \in \mathcal{T}_\Sigma \) which can be derived under the rigidity condition. For a given derivation \( \mathcal{D} : \theta = t_0 \to_G t_1 \to_G \ldots \to_G t_n = t \) and a non-terminal \( \beta \) we say that \( p \in \text{Pos}(t) \) is a \( \beta \)-position in \( \mathcal{D} \) if there is an \( i \leq n \) with \( t_i[p] = \beta \), i.e., either a production rule \( \beta \to s \) has been applied at \( p \) in \( \mathcal{D} \), or \( \beta \) occurs at position \( p \) in \( t \). In the context of a given grammar \( G \), we sometimes write \( \mathcal{D} : \alpha \to_G t \) to specify that \( \mathcal{D} \) is a derivation starting with \( \alpha \) and ending with the term \( t \).

\[ \textbf{Lemma 6.} \text{ Let } G = (N, N_R, \Sigma, \theta, P) \text{ be a rigid tree grammar and let } t \in L(G). \text{ Then there is a derivation } \theta \to_G \ldots \to_G t \text{ which uses at most one } \beta \text{-production for each } \beta \in N_R. \]

\[ \textbf{Proof.} \text{ Given any derivation of } t, \text{ suppose both } \beta \to s_1 \text{ and } \beta \to s_2 \text{ are used at positions } p_1 \text{ and } p_2 \text{ respectively. Then by the rigidity condition } t[p_1] = t[p_2] \text{ and we can replace the derivation at } p_2 \text{ by that at } p_1 \text{ (or the other way round). This transformation does not violate the rigidity condition because it only copies existing parts of the derivation.} \]

\[ \textbf{Lemma 7.} \text{ Let } G = (N, N_R, \Sigma, \theta, P) \text{ be a totally rigid tree grammar and } \theta \neq \beta \in N_R, \text{ such that there is exactly one } t \text{ with } \beta \to t \text{ in } P. \text{ If } G' = (N_R \setminus \{\beta\}, \Sigma, \theta, (P \setminus \{\beta \to t\}) |_{\beta \to t} \text{ then } L(G) = L(G'). \]

\[ \textbf{Proof.} \text{ If a } G \text{-derivation of a term } s \text{ uses } \beta, \text{ it must replace } \beta \text{ by } t \text{ hence } s \text{ is derivable using the productions of } G' \text{ as well. The rigidity condition is preserved as the equality constraints of the } G' \text{-derivation are a subset of those of the } G \text{-derivation. Conversely, given a } G' \text{-derivation of a term } s \text{ we obtain a derivation of } s \text{ from the productions of } G \text{ by replacing applications of } \delta \to r[\beta \setminus t] \text{ by } \delta \to r \text{ followed by a copy of } \beta \to t \text{ for each occurrence of } \beta \text{ in } r. \text{ Let } \gamma_1, \ldots, \gamma_n \text{ be the non-terminals that appear in } t. \text{ By the rigidity condition for } i \in \{1, \ldots, n\} \text{ there is a unique term at all } \gamma_i \text{-positions in the derivation. Hence } \beta \text{ fulfills the rigidity condition as well, and we have obtained a } G \text{-derivation of } s. \]

\[ \textbf{Lemma 8.} \text{ If a rigid tree grammar } G' \text{ is obtained from another rigid tree grammar } G \text{ by deletion of production rules, then } L(G') \subseteq L(G). \]

\[ \textbf{Proof.} \text{ Every } G' \text{-derivation is a } G \text{-derivation.} \]

\[ \textbf{Notation 9.} \text{ For a given non-terminal } \beta \text{ and a term } t, \text{ we will write } \beta \in t \text{ or } t \ni \beta \text{ for denoting that } \beta \text{ occurs in } t. \]

\[ \textbf{Definition 10.} \text{ Let } G \text{ be a tree grammar. A path of } G \text{ is a list } \mathcal{P} \text{ of productions } \alpha_1 \to t_1, \ldots, \alpha_n \to t_n \text{ with } n \geq 1 \text{ and } \alpha_{i+1} \in t_i \text{ for all } i \in \{1, \ldots, n-1\}. \text{ The length of a path is } |\mathcal{P}| = n. \text{ We will also write } \mathcal{P} : \alpha_1 \to t_1 \ni \alpha_2 \to \ldots \ni \alpha_n \to t_n \text{ to denote a path.} \]

For a given path \( \mathcal{P} : \alpha_1 \to t_1 \ni \alpha_2 \to \ldots \ni \alpha_n \to t_n \) we say that \( \alpha_1, \ldots, \alpha_n \) are on the path \( \mathcal{P} \) and write \( \alpha_i \in \mathcal{P} \) for that. We also write \( \mathcal{P} : \alpha_1 \to t_n \) and \( \mathcal{P} : \alpha_1 \to \alpha_n \), if we do not want to explicitly mention the intermediate steps. For a fixed grammar \( G \), we write \( \alpha \to \beta \) to denote that there is a path \( \mathcal{P} \) in \( G \) with \( \mathcal{P} : \alpha \to \beta \).

For a set \( P \) of production rules, we write \( \alpha \prec_P \beta \) (or simply \( \alpha \prec \beta \), when \( P \) is clear from context) if there is a production \( \alpha \to t \) in \( P \) with \( \beta \in t \). We write \( \prec^+ \) for the transitive
closure of $\prec$, and $\prec^*$ for its reflexive, transitive closure. Note that $\alpha \rightarrow \beta$ implies $\alpha \prec^+ \beta$, but not the other way around, since $\beta$ could be a non-terminal with no production $\beta \rightarrow s$ in $P$.

**Definition 11.** A tree grammar $\langle N, \Sigma, \theta, P \rangle$ is called cyclic if $\alpha \prec^+_P \alpha$ for some $\alpha \in N$, and acyclic otherwise.

**Lemma 12.** If $G$ is totally rigid and acyclic, then up to renaming of the non-terminals $G = \langle \{\alpha_1, \ldots, \alpha_n\}, \Sigma, \alpha_1, P \rangle$ with $L(G) = \{\alpha_1[\alpha_1 \backslash t_1] \cdots [\alpha_n \backslash t_n] \mid \alpha_i \rightarrow t_i \in P\}$.

**Proof.** Acyclicity permits a renaming of non-terminals, such that $\alpha_i \prec^+_P \alpha_j$ implies $i < j$. Then $L(G) \supseteq \{\alpha_1[\alpha_1 \backslash t_1] \cdots [\alpha_n \backslash t_n] \mid \alpha_i \rightarrow t_i \in P\}$ is obvious. For the left-to-right inclusion, let $\mathcal{D}: \alpha_0 = s_0 \rightarrow_G \cdots \rightarrow_G s_n = s \in \mathcal{R}_\Sigma$ be a derivation in $G$. By Lemma 6 we can assume that for each $j$ at most one production whose left-hand side is $\alpha_j$ is applied, say $\alpha_j \rightarrow t_j$. By acyclicity we can rearrange the derivation so that $\alpha_j \rightarrow t_j$ is only applied after $\alpha_i \rightarrow t_i$ for all $i < j$. For those $\alpha_j$ which do not appear in the derivation we can insert any substitution without changing the final term so we obtain $s = \alpha_0[\alpha_0 \backslash t_0] \cdots [\alpha_n \backslash t_n]$. ◀

This lemma entails that $|L(G)| \leq \prod_{i=1}^n |\{t \mid \alpha_i \rightarrow t \in P\}|$, in particular we are dealing with a finite language. The central questions in this context are (in contrast to the standard setting in formal language theory) not concerned with representability but with the size of a representation.

### 4 Proofs as Grammars

We will now restrict our attention to a certain class of proofs, called simple proofs below.

**Definition 13.** A proof $\pi$ is called simple if it is regular, the end-sequent is of the form $\exists x_1 \cdots \exists x_n A$ with $A$ quantifier-free, and every cut in $\pi$ whose cut-formula contains a quantifier is of the following form, where $B$ is quantifier-free:

$$
\begin{array}{c}
\Gamma, \exists x B \\
\forall x B, \Delta
\end{array}
\quad \text{cut} \quad (1)
$$

The above definition requires regularity which is a necessary assumption in the context of cut-elimination. The restriction of the end-sequent is done for expository purposes only, and can be extended to arbitrary sequents. The requirement of the $\forall$-rule being applied directly above the cut is natural as the rule is invertible. Moreover, any proof which does not fulfill this requirement can be pruned to obtain one that does, by simply permuting $\forall$-inferences down and identifying their eigenvariables when needed. The only significant restriction is that of disallowing quantifier alternations in the cut formulas. We conjecture that the central results extend to the general case. However, this will require the development of an adequate class of grammars.

**Observation 14.** Simple proofs have the technically convenient property of exhibiting a 1-1 relationship between eigenvariables and cuts. For an eigenvariable $\alpha$ we will therefore write $\forall_\alpha$ for the inference introducing $\alpha$ and $\text{cut}_\alpha$ for the corresponding cut.
Definition 15. Let $\pi$ be a proof of $\exists x_1 \cdots \exists x_n A$ and let $\psi$ be a subproof of $\pi$. The Herbrand-set $H(\psi, \pi)$ of $\psi$ with respect to $\pi$ is defined as follows. If $\psi$ is an axiom, then $H(\psi, \pi) = \emptyset$. If $\psi$ is of the form

$$\Gamma, A[x_1,t] \mid \exists \Gamma, \exists x_1 A$$

then $H(\psi, \pi) = H(\psi', \pi) \cup \{A[x\setminus t]\}$. If $\psi$ ends with any unary inference and $\psi'$ is its immediate subproof then $H(\psi, \pi) = H(\psi', \pi)$. If $\psi$ ends with a binary rule and $\psi'$ and $\psi''$ are its immediate subproofs, then $H(\psi, \pi) = H(\psi', \pi) \cup H(\psi'', \pi)$. We write $H(\pi)$ for $H(\pi, \pi)$.

Definition 16. Let $Q$ be an occurrence of a formula $\exists x A$ in a proof. We define the set $\text{tm}(Q)$ of terms associated with $Q$ as follows: if $Q$ is introduced as the main formula of a weakening, then $\text{tm}(Q) = \emptyset$. If $Q$ is introduced by a quantifier rule $\Gamma, A[x\setminus t] \mid \exists \Gamma, \exists x A$ then $\text{tm}(Q) = \{t\}$. If $Q$ is the main formula in the conclusion of a contraction, and $Q_1$ and $Q_2$ are the two occurrences of the same formula in the premise that are contracted, then $\text{tm}(Q) = \text{tm}(Q_1) \cup \text{tm}(Q_2)$. In all other cases, an inference with the occurrence $Q$ in the conclusion has a corresponding occurrence $Q'$ of the same formula in one of its premises, and we let $\text{tm}(Q) = \text{tm}(Q')$.

Definition 17. Let $\pi$ be a simple proof, let $\alpha \in \text{EV}(\pi)$, and let $Q$ be the occurrence of the existentially quantified cut-formula in the premise of $\text{cut}_\alpha$. Then we write $B(\alpha)$ for the set $\{[\alpha\setminus t] \mid t \in \text{tm}(Q)\}$ of substitutions and we define $B(\pi) = \bigcup_{\alpha \in \text{EV}(\pi)} B(\alpha)$.

Structures similar to the above $B(\pi)$ have been investigated also in [14] and [22] where they form the basis of proof net like formalisms using local reductions for quantifiers in classical first-order logic. Our aim in this work is however quite different: we use these structures for a global analysis of the sequent calculus.

Definition 18. The grammar of a simple proof $\pi$ is defined to be the totally rigid grammar $G(\pi) = \langle N_R, \Sigma, \theta, P \rangle$ with

\[
N_R = \text{EV}(\pi) \cup \{\theta\}\\
\Sigma = \Sigma(\pi) \cup \{\land, \lor\}\\
P = \{\theta \rightarrow A \mid A \in H(\pi)\} \cup \{\alpha \rightarrow t \mid [\alpha\setminus t] \in B(\pi)\}
\]

where $\Sigma(\pi)$ is the signature of $\pi$, the rank of $\land$ and $\lor$ is 2, and $\theta$ does not occur in $\pi$.

Lemma 19. If $\pi$ is a simple proof, then $G(\pi)$ is acyclic.

Proof. By induction on the number of cuts in $\pi$. The grammar of a cut-free proof is trivially acyclic. For the induction step, let $r$ be the lowest binary inference with subproofs $\pi_1$ and $\pi_2$ s.t. either (i) $r$ is a cut or (ii) $r$ is not a cut but both $\pi_1$ and $\pi_2$ contain at least one cut. Let $P_1$ and $P_2$ be the sets of productions induced by the cuts in $\pi_1$, $\pi_2$, respectively. If $\pi$ contains a binary inference, $\alpha_r \in \pi_r$ is acyclic and as the cut-formula in $r$ contains at most one quantifier, also $\alpha_r$ is acyclic. Therefore, a cycle in $\alpha_r^+$ must be of the form $\alpha_1, \beta_1, \alpha_2, \beta_2, \alpha_3, \beta_3, \alpha_4, \beta_4$ where $\alpha_1, \beta_1 \in \text{EV}(\pi_1)$ and $\alpha_2, \beta_2 \in \text{EV}(\pi_2)$. However, $r$ contains only one quantifier and depending on its polarity all productions in $P_r$ lead from $\pi_1$ to $\pi_2$ or from $\pi_2$ to $\pi_1$ but not both, so $\alpha_r$ is acyclic.
We now come to a central definition of this paper.

**Definition 20.** For a simple proof $\pi$, we define its Herbrand-content as $[\pi] = L(G(\pi))$.

Lemma 19 together with Lemma 12 implies that the Herbrand-content of a simple proof $\pi$ with $n$ cuts can be written as

$$[\pi] = \{A[\alpha_1 \backslash t_1] \cdot \cdot \cdot [\alpha_n \backslash t_n] \mid A \in H(\pi), [\alpha_i \backslash t_i] \in B(\alpha_i)\}$$

Note that for cut-free $\pi$ we have $[\pi] = H(\pi)$, i.e. the Herbrand-content is nothing else but the Herbrand-disjunction induced by the proof. Furthermore, the Herbrand-content is a strong invariant: it is not changed by axiom reduction, propositional reduction and inference permutations as those transformations do not change the grammar. Furthermore, Lemma 7 shows that $[\pi]$ is not changed by quantifier reduction and Lemma 8 shows that if $\pi \rightsquigarrow \pi'$ is a step of weakening reduction then $[\pi'] \subseteq [\pi]$. A more difficult result is that the Herbrand-content is even invariant under the reduction of a contraction; the following section is devoted to proving this.

## 5 Invariance under Duplication

For simplifying the presentation, we assume in the following (without loss of generality) that the $\forall$-side is on the right of a cut and the $\exists$-side on the left. Then, a production $\beta \rightarrow t$ in $G(\pi)$ corresponds to three inferences in $\pi$: a cut, an instance of the $\forall$-rule, and an instance of the $\exists$-rule, that we denote by $\text{cut}_\beta$, $\forall_\beta$, and $\exists_\beta$, respectively, and that are, in general, arranged in $\pi$ as shown below.

\[
\begin{align*}
\Gamma', A[x \backslash t] & \vdash \exists x A \\
\Gamma, A & \vdash \forall x \exists x A, \Delta' \\
\vdots & \\
\Gamma, A & \vdash \forall x \exists x A, \Delta \\
\Gamma, \Delta & \vdash \text{cut}_\beta 
\end{align*}
\]  

(2)

The additional condition that $\forall_\beta$ is directly above $\text{cut}_\beta$, as indicated in (1) is only needed because in the following we make extensive use of Observation 14: there is a one-to-one correspondence between the cuts and the eigenvariables in $\pi$, and thus, the notation $\text{cut}_\beta$ makes sense.

Furthermore, we say that the instances $\text{cut}_\beta$, $\forall_\beta$, and $\exists_\beta$ are on a path $\mathcal{P}$ in $G(\pi)$ if the production $\beta \rightarrow t$ is in $\mathcal{P}$.

**Definition 21.** Let $\pi$ be a proof containing the configuration $r_1 \overset{\exists}{\cdots} r_2 \overset{\forall}{\cdots} r_3$, where $r_1, r_2,$ and $r_3$ are arbitrary rule instances, and $r_3$ is a branching rule, and $r_1$ and $r_2$ might or might not be branching. Then we say that $r_1$ is on the left above $r_3$, denoted by $r_1 \upaarrow r_3$, and $r_2$ is on the right above $r_3$, denoted by $r_3 \downarrow r_2$, and $r_1$ and $r_2$ are in parallel, denoted by $r_1 \parallel r_2$.

**Lemma 22.** Let $\pi$ be a simple proof and $\mathcal{P} : \alpha_1 \rightarrow t_1 \exists \alpha_2 \rightarrow t_2 \cdots \rightarrow t_n$ be a path in $G(\pi)$. Then there is a $k \in \{1, \ldots, n\}$ s.t. $\text{cut}_{\alpha_k}$ is lowermost among all inferences on $\mathcal{P}$. Furthermore, $\forall_{\alpha_k}$ is on the right above $\text{cut}_{\alpha_k}$ and $\exists_{\alpha_k}$ is on the left above $\text{cut}_{\alpha_k}$.

**Proof.** We proceed by induction on $n$. If $n = 1$, then $n = k = 1$. For the induction step consider a path $\alpha_1 \rightarrow t_1 \exists \cdots \exists \alpha_n \rightarrow t_n \exists \alpha_{n+1} \rightarrow t_{n+1}$. As $\alpha_{n+1} \in t_n$ we know that $\exists_{\alpha_n}$
must be on the right above \( \text{cut}_{\alpha_{n+1}} \). By induction hypothesis there is a \( k \in \{1, \ldots, n\} \) such that we are in one of the following two situations

\[
\begin{array}{ccc}
\exists_{\alpha_{k+1}} & \forall_{\alpha_k} \\
\text{cut}_{\alpha_{k+1}} & \text{or} & \exists_{\alpha_{n+1}} & \forall_{\alpha_n} \\
\end{array}
\]

In the first case we let \( l = n + 1 \) and in the second we let \( l = k \). In both cases \( \text{cut}_{\alpha_l} \) has the desired properties.

\[\textbf{Lemma 23.} \text{ Let } \pi \text{ be a simple proof, } G(\pi) = (N_R, \Sigma, \theta, P), \text{ and } \beta, \alpha \in \text{EV}(\pi). \text{ If } \beta \rightarrow \alpha \text{ then either } \text{cut}_\alpha \vdash \text{cut}_\beta \text{ or } \text{cut}_\alpha \vdash \text{cut}_\beta \text{ or } \text{cut}_\alpha \vdash \text{cut}_\beta. \]

\[\textbf{Proof.} \text{ Since } \beta \rightarrow \alpha, \text{ we have a path } \beta \rightarrow \ldots \rightarrow \alpha \rightarrow t \text{ for some } t. \text{ By Lemma 22 there is a } \gamma, \text{ such that } \exists_{\gamma} \vdash \text{cut}_\gamma \text{ and } \text{cut}_\gamma \vdash \forall_{\beta}, \text{ and such that } \text{cut}_\alpha \text{ and } \text{cut}_\gamma \text{ are not below } \text{cut}_\gamma. \text{ Furthermore, } \text{cut}_\alpha \text{ must be below } \exists_{\gamma}, \text{ and } \text{cut}_\beta \text{ below } \forall_{\beta}. \text{ If } \gamma = \beta, \text{ then } \text{cut}_\alpha \vdash \text{cut}_\beta. \text{ If } \gamma = \alpha, \text{ then } \text{cut}_\alpha \vdash \text{cut}_\beta. \text{ And if } \gamma \neq \beta \text{ and } \gamma \neq \alpha, \text{ then } \text{cut}_\alpha \vdash \text{cut}_\beta. \]

\[\textbf{Lemma 24.} \text{ Let } G(\pi) = (N_R, \Sigma, \theta, P) \text{ be the grammar of a simple proof } \pi, \text{ such that there are two paths}
\]

\[
\begin{array}{c}
\beta \rightarrow t \\triangleright 0 \rightarrow 0 = 0 \rightarrow 0 \rightarrow 0 = t \in \mathcal{P}
\end{array}
\]

\[
\begin{array}{c}
\beta \rightarrow t \triangleright 0 \rightarrow 0 \rightarrow 0 \rightarrow 0 \rightarrow 0 = \alpha \rightarrow s_n
\end{array}
\]

\[
\begin{array}{c}
\beta \rightarrow t \triangleright 0 \rightarrow 0 \rightarrow 0 \rightarrow 0 \rightarrow 0 = \alpha \rightarrow r_m
\end{array}
\]

such that \( \gamma_0 \) and \( \delta_0 \) occur at two different positions in \( t \). Then we have one of the following two cases:

1. we have \( \gamma_i = \delta_j \) for some \( 0 \leq i < n \) and \( 0 \leq j < m \), or
2. for all \( 0 \leq i < n \) and \( 0 \leq j < m \) we have \( \text{cut}_{\alpha_i} \vdash \text{cut}_{\gamma_i} \) and \( \text{cut}_{\alpha_j} \vdash \text{cut}_{\delta_j} \).

\[\textbf{Proof.} \text{ Note that because of acyclicity of } G(\pi), \text{ we have that } \beta \neq \gamma_i \text{ for all } i \leq n \text{ and } \beta \neq \delta_j \text{ for all } j \leq m, \text{ in particular } \beta \neq \alpha. \text{ Assume, for the moment, that } m, n > 0; \text{ the case of one of them being zero will be treated at the very end of the proof. Then } \gamma_0 \neq \alpha \text{ and } \delta_0 \neq \alpha. \text{ If } \gamma_0 = \delta_0, \text{ we have case 1. So, assume also } \gamma_0 \neq \delta_0. \text{ As } \beta \rightarrow t \text{ is a production in } G(\pi), \text{ the proof } \pi \text{ contains a formula which contains both } \gamma_0 \text{ and } \delta_0 \text{ hence } \forall_{\gamma_0} \text{ and } \forall_{\delta_0} \text{ are not parallel. Since we have } \text{cut}_{\gamma_0} \vdash \forall_{\gamma_0} \text{ and } \text{cut}_{\delta_0} \vdash \forall_{\delta_0}, \text{ we also have that } \text{cut}_{\gamma_0} \text{ and } \text{cut}_{\delta_0} \text{ are not parallel. Without loss of generality, assume that } \text{cut}_{\delta_0} \text{ is below } \text{cut}_{\gamma_0}. \text{ Then } \text{cut}_{\delta_0} \vdash \text{cut}_{\gamma_0} \text{ (since } \text{cut}_{\gamma_0} \vdash \text{cut}_{\delta_0} \text{ would entail } \forall_{\gamma_0} \triangleright \forall_{\delta_0}). \text{ Since we have } \delta_0 \rightarrow \alpha, \text{ we can apply Lemma 23, giving us three possibilities:}
\]

\[
\begin{array}{ccc}
\exists_{\alpha_i} & \forall_{\alpha_i} & \exists_{\delta_0} \\
\text{cut}_{\alpha_i} & \text{cut}_{\delta_0} & \text{cut}_{\gamma_0}
\end{array}
\]

By Lemma 22 applied to the path \( \gamma_0 \rightarrow s_n \) we have that \( \text{cut}_{\delta_0} \) must coincide with \( \text{cut}_{\gamma_i} \) for some \( 0 \leq i < n \) (since \( \pi \) is a tree), so \( \delta_0 = \gamma_i \) (by Observation 14), and we are in case 1.
If \( \text{cut}_\alpha \vdash \text{cut}_\delta \) then we are in both of the following two situations:

\[
\begin{array}{c}
\exists s_n \\
\text{cut}_{s_n} \\
\vdash \forall_{\gamma_0} \\
\end{array}
\quad \text{and} \quad
\begin{array}{c}
\exists r_m \\
\text{cut}_{r_m} \\
\vdash \forall_{\delta_0} \\
\end{array}
\]

Thus, by Lemma 22 applied to the paths \( \gamma_0 \rightarrow \ldots \rightarrow s_n \) and \( \delta_0 \rightarrow \ldots \rightarrow r_m \), we know that \( \text{cut}_\alpha = \text{cut}_{\gamma_i} = \text{cut}_{\delta_i} \) for some \( 0 \leq k \leq n \) and \( 0 \leq l \leq m \) hence \( \gamma_k = \alpha = \delta_l \). Furthermore \( k = n \) and \( l = m \) by acyclicity of \( G(\pi) \). Now consider any \( \gamma_i \) with \( 0 \leq i < n \). Since \( \gamma_i \rightarrow \alpha \), we can apply Lemma 23 and get either \( \text{cut}_\alpha \vdash \text{cut}_{\gamma_i} \) or \( \text{cut}_\alpha \vdash \text{cut}_{\gamma_i} \). Since by Lemma 22 \( \text{cut}_{\gamma_i} \) must be above \( \text{cut}_\alpha \), we conclude \( \text{cut}_\alpha \vdash \text{cut}_{\gamma_i} \). With the same reasoning we can conclude that \( \text{cut}_\alpha \vdash \text{cut}_{\delta_j} \) for all \( 0 \leq j < m \). We are therefore in case 2.

If \( \text{cut}_\alpha \vdash \text{cut}_\delta \) then we are in both of the following two situations:

\[
\begin{array}{c}
\exists s_n \\
\text{cut}_{s_n} \\
\vdash \forall_{\gamma_0} \\
\end{array}
\quad \text{and} \quad
\begin{array}{c}
\exists r_m \\
\text{cut}_{r_m} \\
\vdash \forall_{\delta_0} \\
\end{array}
\]

By Lemma 22 applied to the paths \( \gamma_0 \rightarrow \ldots \rightarrow s_n \) and \( \delta_0 \rightarrow \ldots \rightarrow r_m \), the rule \( r \) coincides with \( \text{cut}_{\gamma_i} \) and \( \text{cut}_{\delta_j} \) for some \( 0 < i < n \) and \( 0 < j < m \), therefore \( \gamma_i = \delta_j \) (by Observation 14), and we are in case 1.

It remains to treat the case \( n = 0 \) or \( m = 0 \). If \( m = n = 0 \) then we are trivially in case 2 (there is no \( 0 \leq i < n \) or \( 0 \leq j < m \)). If \( n = 0 \) and \( m > 0 \), we can apply Lemma 22 to the path \( \delta_0 \rightarrow \ldots \rightarrow r_m \) and obtain an \( l \in \{0, \ldots, m\} \) such that we are in the situation

\[
\begin{array}{c}
\exists r_m \\
\text{cut}_{r_m} \\
\vdash \forall_{\delta_0} \\
\end{array}
\]

But by the same argument as at the beginning of the proof, we also have that \( \forall_{\alpha} \) and \( \forall_{\delta_0} \) cannot be in parallel (\( \alpha \) and \( \delta_0 \) both appear in \( \tau \)), and therefore either \( \text{cut}_{\delta_0} \vdash \text{cut}_\alpha \) or \( \text{cut}_\alpha \vdash \text{cut}_{\delta_0} \). Since \( \delta_0 \rightarrow \alpha \), the only possibility is \( \text{cut}_\alpha \vdash \text{cut}_{\delta_0} \), by Lemma 23. Thus \( \text{cut}_\alpha = \text{cut}_{\delta_1} \), and therefore \( l = m \) and we are in case 2. The case \( m = 0 \) and \( n > 0 \) is similar.

The following is the main result of this section:

**Proposition 25.** Let \( \pi \) be a simple proof that contains a subproof \( \psi \), shown on the left below,

\[
\psi = \frac{\psi_1}{\Gamma, A, A} \quad \frac{\psi_2}{\Gamma, A} \quad \text{cut} = \frac{\Gamma, A, A}{\Delta, \Delta} \quad \text{cut} = \psi'
\]

\[
\frac{\psi_1}{\Gamma, A, A} \quad \frac{\psi_2}{\Gamma, A} \quad \text{cut} = \frac{\Gamma, A, A}{\Delta, \Delta} \quad \text{cut} = \psi'
\]

\[
\frac{\psi_1}{\Gamma, A, A} \quad \frac{\psi_2}{\Gamma, A} \quad \text{cut} = \frac{\Gamma, A, A}{\Delta, \Delta} \quad \text{cut} = \psi'
\]

\[
\frac{\psi_1}{\Gamma, A, A} \quad \frac{\psi_2}{\Gamma, A} \quad \text{cut} = \frac{\Gamma, A, A}{\Delta, \Delta} \quad \text{cut} = \psi'
\]
Herbrand-Confluence for Cut Elimination in Classical First Order Logic

and let $\pi'$ be the proof obtained from $\pi$ from replacing $\psi$ by $\psi'$ shown on the right above, where $\rho' = [\alpha' \alpha'']_{a \in \text{EV}(\psi_2)}$ and $\rho'' = [\alpha' \alpha'']_{a \in \text{EV}(\psi_2)}$ are substitutions that replace all eigenvariables in $\psi_2$ by fresh copies. Then $[\pi] = [[\pi']].$

**Proof.** Let us first show $[[\pi']] \subseteq [\pi]$: write $P$ for the productions of $G(\pi)$ and $P'$ for those of $G(\pi')$. Let $F \in [\pi] = L(G(\pi))$ and $D$ be its derivation. If the duplicated cut is quantifier-free, then $P' = P\rho' \cup P\rho''$ hence $D\rho'$ (as well as $D\rho''$) is a derivation of $F$ in $G(\pi')$. If the duplicated cut contains a quantifier, let $\alpha$ be its eigenvariable, let $t_1, \ldots, t_n$ be its terms coming from the left copy of $A$ and $t_{k+1}, \ldots, t_m$ those from the right copy of $A$ and let $Q = \{ \alpha \rightarrow t_1, \ldots, \alpha \rightarrow t_n \} \subseteq P$. We then have

$$P' = (P \setminus Q)\rho' \cup \{ \alpha' \rightarrow t_1, \ldots, \alpha' \rightarrow t_k \} \cup (P \setminus Q)\rho'' \cup \{ \alpha'' \rightarrow t_{k+1}, \ldots, \alpha'' \rightarrow t_m \}.$$ 

If $D$ does not contain $\alpha$, then $D\rho'$ (as well as $D\rho''$) is a derivation of $F$ in $G(\pi')$. If $D$ contains $\alpha$, then by Lemma 6 we can assume that it uses only one $\alpha$-production, say $\alpha \rightarrow t_i$. If $1 \leq i \leq k$, then $D\rho'$ is a derivation of $F$ in $G(\pi')$ and if $k < i \leq n$, then $D\rho''$ is a derivation of $F$ in $G(\pi')$.

Let us now show $[[\pi']] \subseteq [\pi]$: let $F$ be a formula in $[[\pi']] = L(G(\pi'))$, and let $D'$ be a derivation of $F$ in $G(\pi')$. We construct $D = D'(\rho')^{-1}(\rho'')^{-1}$ by "undoing" the renaming of the variables in $\psi_2$. Then $D$ is a derivation for $F$, using the production rules of $G(\pi)$, but possibly violating the rigidity condition.

First, observe that only non-terminals $\alpha \in \text{EV}(\psi_2)$ can violate the rigidity condition in $D$: if $\beta \notin \text{EV}(\psi_2)$ violates the rigidity condition then there are $\beta$-positions $p_1, p_2$ in $D$ with $F|_{p_1} \neq F|_{p_2}$ and as $\beta\rho' \rho'' = \beta$ the positions $p_1, p_2$ are also $\beta$-positions in $D'$ and they violate the rigidity condition in $D'$ which is a contradiction to $D'$ being a $G(\pi')$-derivation.

Now define for each $\alpha \in \text{EV}(\psi_2)$ the value $n(D, \alpha)$ to be the number of pairs $(p_1, p_2) \in \text{Pos}(F) \times \text{Pos}(F)$ where $p_1$ and $p_2$ are $\alpha$-positions in $D$ with $p_1 \neq p_2$ and $F|_{p_1} \neq F|_{p_2}$, and define $n(D, \alpha) = \sum_{\alpha \in \text{EV}(\psi_2)} n(D, \alpha)$. We proceed by induction on $n(D)$ to show that $D$ can be transformed into a derivation which does no longer violate rigidity. If $n(D) = 0$ then $D$ obeys the rigidity condition, and we are done. Otherwise there is at least one $\alpha \in \text{EV}(\psi_2)$ with $n(D, \alpha) > 0$. We now pick one such $\alpha$ which is minimal with respect to $\prec^*$ (which exists since $G(\pi)$ is acyclic). Let $p_1$ and $p_2$ be $\alpha$-positions in $D$ with $p_1 \neq p_2$ and $F|_{p_1} \neq F|_{p_2}$, let $p$ be the maximal common prefix of $p_1$ and $p_2$ and let $q$ be the maximal prefix of $p$ where a production rule has been applied in $D$. Due to the tree structure of $F$, the position $q$ is uniquely defined, and $q$ is a $\beta$-position for some non-terminal $\beta$, and some production rule $\beta \rightarrow t$ has been applied at position $q$ in $D$, and we have two paths:

- $\beta \rightarrow t \ni \gamma_0 \rightarrow s_0 \ni \gamma_1 \rightarrow s_1 \ni \ldots \rightarrow s_{n-1} \ni \gamma_n = \alpha \rightarrow s_n$
- $\beta \rightarrow t \ni \delta_0 \rightarrow r_0 \ni \delta_1 \rightarrow r_1 \ni \ldots \rightarrow r_{m-1} \ni \delta_m = \alpha \rightarrow r_m$

where $\gamma_0$ and $\delta_0$ occur at two different positions in $t$. Thus, we can apply Lemma 24, giving us the following two cases:

- We have $\gamma_i = \delta_j$ for some $0 \leq i < n$ and $0 \leq j < m$. Say $\eta = \gamma_i = \delta_j$, and let $p_\gamma$ and $p_\delta$ be the positions of $\gamma_i$ and $\delta_j$ (respectively) in $D$. Since $\eta \prec^+$ $\alpha$ we know that $\eta$ does not violate the rigidity condition (we chose $\alpha$ to be minimal), and therefore $F|_{p_\gamma} = F|_{p_\delta} = F'$. Let $D_1 : \gamma_i \rightarrow \gamma_{G(\pi)} F' \ni \eta \rightarrow \eta_{G(\pi)} F'$. Then $D_1$ and $D_2 : \delta_j \rightarrow \delta_{G(\pi)} F'$ be the two subderivations of $D$ starting in positions $p_\gamma$ and $p_\delta$, respectively. Without loss of generality, we can assume that $n(D_1) \leq n(D_2)$. Then let $D$ be the derivation obtained from $D$ by replacing $D_2$ by $D_1$. Thus $D$ is still a derivation for $F$, but $n(D) < n(D_2)$.

- For all $0 \leq i < n$ and $0 \leq j < m$ we have cut$_\eta^r$ cut$_\gamma$ and cut$_\eta^r$ cut$_\delta$. So all inferences of the path $\gamma_0 \rightarrow \ldots \rightarrow s_{n-1}$ as well as all inferences of $\delta_0 \rightarrow \ldots \rightarrow r_{m-1}$ are in $\psi_2$. Therefore all variables of the paths are in $\text{EV}(\psi_2)$. As $\alpha$ violates the rigidity in $D$
one of \( p_1, p_2 \) must be a \( \alpha' \)-position and the other a \( \alpha'' \)-position in \( \mathcal{D}' \) because \( \mathcal{D}' \) does satisfy the rigidity condition. Without loss of generality we can assume that \( p_1 \) is the \( \alpha' \)-position and \( p_2 \) the \( \alpha'' \)-position. As the paths are contained completely in \( \psi_2 \) we have \( \gamma_0 \in \text{EV}(\psi_2)\rho' \) and \( \delta_0 \in \text{EV}(\psi_2)\rho'' \) which is a contradiction as no term can contain both a variable from \( \text{EV}(\psi_2)\rho' \) and one from \( \text{EV}(\psi_2)\rho'' \).

6 Herbrand-Confluence

We now turn to cut reduction sequences that start with a simple proof. All the reductions shown in Figure 1 preserve simplicity, except the following:

\[
\begin{align*}
\cdots & \quad \forall \alpha & \quad \forall \beta & \quad \exists & \quad \forall \alpha & \quad \forall \beta \\
\cdots & \quad \text{cut}_\alpha & \quad \text{cut}_\beta & \quad \Rightarrow & \quad \text{cut}_\alpha & \quad \text{cut}_\beta
\end{align*}
\]

where \( \text{cut}_\alpha \) is permuted down under \( \text{cut}_\beta \) (using the bottommost reduction in Fig. 1) and the cut formula of \( \text{cut}_\beta \) has its ancestor on the right side of \( \text{cut}_\alpha \). So in the following, when we speak about a reduction sequence of simple proofs we require that the above reduction is immediately followed by permuting \( \forall \alpha \) down as well, in order to arrive at

\[
\begin{align*}
\cdots & \quad \forall \beta & \quad \forall \alpha \\
\cdots & \quad \text{cut}_\beta & \quad \text{cut}_\alpha
\end{align*}
\]

which is again simple. Recall that for our result this step is not strictly needed. We only add it here to simplify the presentation.

Collecting together the results proved in this paper we then obtain the following theorem.

\begin{proposition}
If \( \pi \rightsquigarrow \pi' \) is a reduction sequence of simple proofs, then \( \llbracket \pi \rrbracket \supseteq \llbracket \pi' \rrbracket \).
\end{proposition}

\begin{proof}
By induction on the length of the reduction \( \pi \rightsquigarrow \pi' \) making a case distinction on the applied reduction step. If \( \pi_i \rightsquigarrow \pi_{i+1} \) is a propositional reduction, an axiom reduction or a rule permutation, we even have \( G(\pi_i) = G(\pi_{i+1}) \). If it is a quantifier reduction, then \( \llbracket \pi_i \rrbracket = \llbracket \pi_{i+1} \rrbracket \) by Lemma 7. If it is the reduction of a contraction, then \( \llbracket \pi_i \rrbracket = \llbracket \pi_{i+1} \rrbracket \) by Proposition 25. If it is the reduction of a weakening, then \( \llbracket \pi_i \rrbracket \supseteq \llbracket \pi_{i+1} \rrbracket \) by Lemma 8.
\end{proof}

\begin{corollary}
If \( \pi \rightsquigarrow \pi' \) is a reduction sequence of simple proofs and \( \pi' \) is cut-free, then \( H(\pi') \subseteq \llbracket \pi \rrbracket \).
\end{corollary}

This corollary shows that \( \llbracket \pi \rrbracket \) is an upper bound (w.r.t. the subset relation) on the Herbrand-disjunctions obtainable by cut-elimination from \( \pi \). Let us now compare this result with another upper bound that has previously been obtained in [16]. To that aim let \( G_0(\pi) \) denote the regular tree grammar underlying \( G(\pi) \) which can be obtained by setting all non-terminals to non-rigid. In this notation, a central result of [16], adapted to this paper’s setting of proofs of non-prenex formulas, is

\begin{proposition}
If \( \pi \rightsquigarrow \pi' \) and \( \pi' \) is cut-free, then \( H(\pi') \subseteq L(G_0(\pi)) \).
\end{proposition}

While the above theorem 28 applies also to non-simple proofs, Corollary 27 is stronger in several respects:

First, the size of the Herbrand-content is by an exponential smaller than the size of the bound given by Theorem 28. Indeed, it is a straightforward consequence of Lemma 12 that
the language of a totally rigid acyclic tree grammar with \( n \) production rules is bound by \( n^n \).

On the other hand, there are acyclic regular tree grammars \( G_n \) with \( 2^n \) productions and \( |L(G_n)| = n^n \) (by encoding in \( G_n \) the construction of a tree of depth \( n \) and branching degree \( n \) with an independent choice between \( n \) constant symbols at each leaf). These grammars can be obtained from appropriately constructed proofs.

Secondly, the class of totally rigid acyclic tree grammars can be shown to be in exact correspondence with the class of simple proofs in the following sense. Not only can we use a totally rigid acyclic tree grammar to simulate the process of cut-elimination, we can also—in the other direction—use cut-elimination to simulate the process of calculating the language of a grammar. It is shown in [17] how to transform an arbitrary acyclic totally rigid tree grammar \( G \) into a simple proof that has a \( \rightsquigarrow \) normal form whose Herbrand-disjunction is essentially the language of \( G \).

The third and—for the purposes of this paper—most important difference is that the bound of Corollary 27 is tight (in a sense that we are going to make precise now). This property of the Herbrand-content leads naturally to a confluence result for classical logic.

For tightening this bound, a first obvious observation is that there is no mechanism for deletion in the grammar but there is one in cut-elimination: the reduction of weakening. So, any cut-elimination strategy that does exactly compute \( \llbracket \pi \rrbracket \) must be non-erasing. Consequently we define the non-erasing cut-reduction \( \rightsquigarrow_{nc} \) as \( \rightsquigarrow \) without the reduction rule for weakening. Note that a \( \rightsquigarrow_{nc} \)-normal form \( \pi \) is an analytic proof as well, e.g. \( H(\pi) \) is a (tautological!) Herbrand-disjunction. In contrast to a \( \rightsquigarrow \)-normal form (which might contain implicit redundancy) a \( \rightsquigarrow_{nc} \)-normal form might also contain explicit redundancy in the form of cuts whose cut-formulas are introduced by weakening on one or on both sides. Non-erasing reduction is also of interest in the context of the \( \lambda \)-calculus where it is often considered in the form of the \( \lambda I \)-calculus and gives rise to the conservation theorem (see Theorem 13.4.12 in [7]). Our situation here is however quite different: neither \( \rightsquigarrow \) nor \( \rightsquigarrow_{nc} \) is confluent and neither of them is strongly normalizing. Nevertheless we obtain:

\[ \text{▶ Theorem 29. If } \pi \rightsquigarrow_{nc} \pi' \text{ is a reduction sequence of simple proofs, then } \llbracket \pi \rrbracket = \llbracket \pi' \rrbracket. \]

\[ \text{Proof. Inspection of the proof of Theorem 26 shows that the reduction of weakening is the only step that does not preserve the Herbrand-content.} \]

\[ \text{▶ Definition 30 (Herbrand-confluence). A relation } \rightsquigarrow \text{ on a set of proofs is called Herbrand-confluent iff } \pi \rightsquigarrow \pi_1 \text{ and } \pi \rightsquigarrow \pi_2 \text{ with } \pi_1 \text{ and } \pi_2 \text{ being normal forms for } \rightsquigarrow \text{ implies that } H(\pi_1) = H(\pi_2). \]

\[ \text{▶ Corollary 31. The relation } \rightsquigarrow_{nc} \text{ is Herbrand-confluent on the set of simple proofs.} \]

How do these results fit together with \( \rightsquigarrow_{nc} \) being neither confluent nor strongly normalizing? In fact, note that it is possible to construct a simple proof which permits an infinite \( \rightsquigarrow_{nc} \) reduction sequence from which one can obtain normal forms of arbitrary size by bailing out from time to time. This can be done by building on the propositional double-contraction example found e.g. in [11, 12, 26] and in a similar form in [28]. While these infinitely many normal forms do have pairwise different Herbrand-disjunctions when regarded as multisets, Corollary 31 shows that as sets they are all the same. This observation shows that the lack of strong normalization is taken care of by using sets instead of multisets as data structure. But what about the lack of confluence? Results like [2] and [15] show that the number of \( \rightsquigarrow \) normal forms with different Herbrand-disjunctions can be enormous. On the other hand we have just seen that \( \rightsquigarrow_{nc} \) induces only a single Herbrand-disjunction: \( \llbracket \pi \rrbracket \). The relation between \( \llbracket \pi \rrbracket \) and the many Herbrand-disjunctions induced by \( \rightsquigarrow \) is explained by Corollary 27: \( \llbracket \pi \rrbracket \) contains them all as subsets.
7 Conclusion

We have shown that non-erasing cut-elimination for the class of simple proofs is Herbrand-confluent. While there are different and possibly infinitely many normal forms, they all induce the same Herbrand-disjunction. This result motivates the definition of this unique Herbrand-disjunction as Herbrand-content of the proof with cut.

As future work, the authors plan to extend this result to arbitrary first-order proofs. The treatment of blocks of quantifiers is straightforward: the rigidity condition must be changed to apply to vectors of non-terminals. Treating quantifier alternations is more difficult: the current results suggest to use a stack of totally rigid tree grammars, each layer of which corresponds to one layer of quantifiers (and is hence acyclic). Concerning further generalizations, note that the method of describing a cut-free proof by a tree language is applicable to any proof system with quantifiers that has a Herbrand-like theorem, e.g., even full higher-order logic as in [23]. The difficulty consists in finding an appropriate type of grammars.

Given the wealth of different methods for the extraction of constructive content from classical proofs, what we learn from our work is this: the first-order structure possesses (in contrast to the propositional structure) a unique and canonical unfolding. The various extraction methods hence do not differ in the choice of how to unfold the first-order structure but only in choosing which part of it to unfold. We therefore see that the effect of the underspecification of algorithmic detail in classical logic is redundancy.

Acknowledgments

The authors would like to thank Paul-André Melliès for helpful comments on this work. The first author was supported by a Marie Curie Intra European Fellowship within the 7th European Community Framework Programme and by the projects I-603 N18 and P22028 of the Austrian Science Fund (FWF).

References

Herbrand-Confluence for Cut Elimination in Classical First Order Logic


