A Clausal Approach to Proof Analysis in Second-Order Logic *

Stefan Hetzl[1], Alexander Leitsch[1], Daniel Weller[1], and Bruno Woltzenlogel Paleo[1]

{hetzl, leitsch, weller, bruno}@logic.at
Institute of Computer Languages (E185),
Vienna University of Technology,
Favoritenstraße 9, 1040 Vienna, Austria

Abstract. This work defines an extension CERES^2 of the first-order cut-elimination method CERES to the subclass of sequent calculus proofs in second-order logic using quantifier-free comprehension. This extension is motivated by the fact that cut-elimination can be used as a tool to extract information from real mathematical proofs, and often a crucial part of such proofs is the definition of sets by formulas. This is expressed by the comprehension axiom scheme, which is representable in second-order logic. At the core of CERES^2 lies the production of a set of clauses CL(φ) from a proof φ that is always unsatisfiable. From a resolution refutation γ of CL(φ), a proof without essential cuts can be constructed. The main theoretical obstacle in the extension of CERES to second-order logic is the construction of this proof from γ. This issue is solved for the subclass considered in this paper. Moreover, we discuss the problems that have to be solved to extend CERES^2 to the complete class of second-order proofs. Finally, the method is applied to a simple mathematical proof that involves induction and comprehension and the resulting proof is analyzed.

1 Introduction

The discipline of *proof mining* deals with the extraction of information from formal proofs. Different methods have been applied successfully (see [1], [2]). This work considers the approach of using (partial) cut-elimination to extract hidden information from proofs.

The first-order cut-elimination method CERES (cut-elimination by resolution) has several advantages over the traditional reductive cut-elimination methods: firstly, the reductive methods are subsumed by CERES (i.e. every proof obtained by a reductive method can also be obtained by CERES, see [2]), and secondly, a non-elementary speed-up over Gentzen's method by the use of CERES is possible (see [3]). The CERES method has been implemented in the `CERES` system[1]. In this paper, we present the extension of CERES to CERES^2, a cut-elimination

* Supported by the Austrian Science Fund (project P19875)

[1] `http://www.logic.at/ceres`

method for second-order logic, which is based on a set of clauses that is extracted from a proof with cuts, the *characteristic clause set*.

The benefits of CERES^2 over traditional cut-elimination methods are twofold: Firstly, the characteristic clause set can be regarded as the kernel of the proof with cuts and as such can provide valuable information that a human could not easily read off of a formal proof (for some evidence supporting this, see [4] and [5]). Secondly, due to the use of a resolution calculus at the core of CERES^2, theoretical and practical advances in higher-order theorem proving may enhance the power of the method.

An inherent limitation of the CERES method (and indeed of all first-order cut-elimination procedures) is that proofs that use comprehension cannot be handled in a straightforward way, as comprehension is essentially a second-order property. In CERES^2, we will be able to handle such proofs in a natural way. The subclass of proofs we are considering here is the class of proofs where comprehension is restricted to quantifier-free formulas. This choice is motivated in part by the fact that converting a resolution refutation to a sequent calculus proof in the presence of arbitrary comprehension (and, therefore, skolemization) is problematic. Indeed, it turns out that, in CERES^2, the construction of proof projections is a highly complicated matter, in contrast to the first-order case.

Note that, due to lack of space, detailed proofs are not developed here, but can be found in [6].

2 The second-order language

Here, we consider a monadic second-order logic based on Church's simply typed λ-calculus [7] and fix the set of *base types* $BT := \{\iota, o\}$, where ι denotes the type of individuals and o the boolean type. The set $\mathcal{T}$ of *types* is built in the usual inductive way over the BT. In contrast to the second-order logic as defined in e.g. [8], we include in the language more objects of order ≤ 2 to allow skolemization, although quantification is restricted to individuals and unary predicates on individuals (i.e. variables of types ι and $\iota \to o$).

We assume given a set of symbols S together with a function $\tau : S \mapsto \mathcal{T}$ assigning types to symbols, where S can be partitioned into the sets V (individual variables), CS (individual constants), FS^n (function symbols), PC^n (predicate constants), PV (unary predicate variables) s.t.

1. For all $x \in V$, $\tau(x) = \iota$,
2. for all $c \in CS$, $\tau(c) = \iota$,
3. for all $f \in FS^n$, $n \geq 1$: $\tau(f) = t_1 \to \ldots \to t_n \to \iota$ where for $1 \leq i \leq n$, $t_i = \iota$ or $t_i = \iota \to o$,
4. for all $P \in PC^n$, $n \geq 0$: $\tau(P) = t_1 \to \ldots \to t_n \to o$ where for $1 \leq i \leq n$, $t_i = \iota$ or $t_i = \iota \to o$,
5. for all $X \in PV$, $\tau(X) = \iota \to o$.

We additionally require that each member of this partition is countably infinite. We define $PC := \bigcup_{i \geq 0} PC^i$ and $FS := \bigcup_{i \geq 1} FS^i$. The set of *expressions* $\mathcal{E}$

is defined inductively in the usual way over the set of symbols together with the symbols $\neg, \wedge, \vee, \rightarrow, \exists, \forall, \lambda, ., (,)$ (keeping in mind the restriction on the order of the types and on the types of the quantified variables). We use infix notation for familiar function symbols and predicates (e.g. $+, =$).

Definition 1. *The* set of second-order formulas *or simply* formulas $SOF := \{F \mid F \in \mathcal{E}, \tau(F) = o\}$. *If* $F \in SOF$, $F \equiv P(t_1, \ldots, t_n)$, $P \in PC \cup PV$, *then* F *is called* atomic.

For atomic formulas $P(t_1, \ldots, t_n)$ we may also write $t_1 \in P(t_2, \ldots, t_n)$. We define the set of *lambda terms* $LT := \{t \mid t \in \mathcal{E}, t \equiv \lambda x.F, \tau(F) = o\}$ and the set of *terms* $T := \{t \mid t \in \mathcal{E}, \tau(t) = \iota\}$. *Polarity* of subexpressions w.r.t. formulas and sequents, *strong* and *weak* quantifiers, the *scope* of quantifiers, *closed* formulas, β-reduction are defined as usual. We assume a variable convention (i.e. variables are renamed appropriately to avoid conflicts).

As proof system for the input and output proofs of the CERES^2 method, we use the sequent calculus **LKDe^2**. This calculus is based on **LK^2** as defined in [9], which consists of the usual structural, propositional, and first-order rules together with second-order quantifier introduction rules that incorporate comprehension. **LKDe^2** extends **LK^2** by rules for first-order equality handling:

$$\frac{\Gamma \vdash \Delta, s = t \quad \Pi \vdash \Lambda, A[s]}{\Gamma, \Pi \vdash \Delta, \Lambda, A[t]} =: r_1 \qquad \frac{\Gamma \vdash \Delta, t = s \quad \Pi \vdash \Lambda, A[s]}{\Gamma, \Pi \vdash \Delta, \Lambda, A[t]} =: r_2$$

LKDe^2 also includes the rules $=: l_1$ and $=: l_2$, and rules for the introduction of definitions (for details, see [10]). All the rules in **LKDe^2** are multiplicative. As axioms we allow the usual tautological sequents $A \vdash A$ for an atomic formula A as well as arbitrary atomic sequents without second-order variables (which is useful for conveniently axiomatizing a background theory). Additionally, if $\mathcal{C}$ is a set of atomic sequents, then we say that π is *an* **LKDe^2***-proof from* $\mathcal{C}$ if for every initial sequent S of π, S is either an axiom, or S is in $\mathcal{C}$. As an intermediary calculus for the construction of a resolution refutation, we use the resolution calculus discussed in the next section.

3 The second-order resolution calculus

In this section, we briefly present the resolution calculus we will need for the CERES^2 method. Note that in second-order logic, in contrast to first-order logic, clauses are not closed under substitution, so the transformation of a formula to clause form has to be incorporated into the calculus, instead of being used just in a preprocessing step.

To use a resolution calculus with CERES^2, it must be possible to use the resolution refutation of a particular set of clauses (the characteristic clause set, see Section 4) as the skeleton of an **LKDe^2**-proof that contains no non-atomic cuts. Intuitively, the following requirements arise:

1. Only literals (i.e. atomic formulas and their negations) may be resolved.

2. It must be possible to produce a propositional resolution refutation from instances of the refuted set of clauses.

Requirement 1 stems from the fact that CERES^2 is a cut-elimination method, and the resolution rule will be translated to the cut rule in **LKDe^2**. Requirement 2 is due to the fact that substitution is integrated in the resolution calculus, while this is not the case with **LKDe^2**.

The resolution calculus we are considering here is a restricted version of the higher-order resolution calculus defined by P.B. Andrews in [11].

Definition 2. *We define a* clause *as a sequent* $C := A_1, \ldots, A_n \vdash B_1, \ldots, B_m$ *with* A_i, B_i *atomic.*

In this paper, the transformation to conjunctive normal form (CNF) is the standard transformation that preserves logical equivalence.

Definition 3. *Let* F *be a quantifier-free formula. Let, modulo commutativity and associativity of* $\vee$, $\mathrm{CNF}(F) \equiv (\neg A_1^1 \vee \ldots \vee \neg A_{k_1}^1 \vee B_1^1 \vee \ldots \vee B_{l_1}^1) \wedge \ldots \wedge (\neg A_1^n \vee \ldots \vee \neg A_{k_n}^n \vee B_1^n \vee \ldots \vee B_{l_n}^n)$. *For* $i \in \{1, \ldots, n\}$, *define the atomic sequent* $C_i \equiv A_1^i, \ldots, A_{k_i}^i \vdash B_1^i, \ldots, B_{l_i}^i$. *Then the* clause form *of* F *is defined as the set* $\{C_1, \ldots, C_n\}$.

Let $S \equiv F_1, \ldots, F_n \vdash G_1, \ldots, G_m$ *be a quantifier-free sequent, then the clause form of* S *is defined as the clause form of* $(F_1 \wedge \ldots \wedge F_n) \rightarrow (G_1 \vee \ldots \vee G_m)$.

A substitution is a pair of mappings: The first maps variables to terms, while the second maps predicate variables to lambda terms. The result of the application of a substitution σ to an expression e is e after replacing all variables by the respective terms and all predicate variables by the respective lambda terms and reducing to β-normal form, this will be denoted by $e\sigma$. A substitution is called quantifier-free if all the (lambda-)terms are quantifier-free.

Definition 4. *We define the application of a quantifier-free substitution* σ *to a set of clauses* $\mathcal{C} = \{C_1, \ldots, C_n\}$, *denoted* $\mathcal{S}(\mathcal{C}, \sigma)$, *as the clause form of the set of quantifier-free sequents* $\{C_1\sigma, \ldots, C_n\sigma\}$. *Note that this includes transformation to CNF, therefore* $|\mathcal{S}(\mathcal{C}, \sigma)| \geq |\mathcal{C}|$.

With this definition, we can state the rules of our resolution calculus.

Definition 5. *In the following,* C, D *are clauses.*

1. C *is called* instance *of* D *if there exists a quantifier-free substitution* σ *s.t.* $C \in \mathcal{S}(\{D\}, \sigma)$.
2. C *is called* p-reduct *of* D *if* C *is* D *after omission of some multiply occuring atomic formulas on either side of the sequent.*
3. *Let* L *be an atom formula,* $C \equiv \Gamma, L \vdash \Delta$ *and* $D \equiv \Gamma' \vdash L, \Delta'$, *then the clause* $\Gamma, \Gamma' \vdash \Delta, \Delta'$ *is called a* resolvent *of* $\{C, D\}$.

Note that we defined resolution without the principle of most general unification (mgu). While the mgu-principle is vital to proof search, the proof transformations in CERES and CERES^2 require resolution proofs after application of global

unifiers. Of course, this definition does not exclude the use of mgu-based provers in the phase of proof search.

Additionally, we use paramodulation rules that allow equality reasoning on the term level. The paramodulation rules are just the restrictions of the equational rules of **LKDe²** to atomic sequents. With this, we can define the notion of a deduction in this calculus:

Definition 6. *Let $\mathcal{C}$ be a set of clauses and let C be a clause. A sequence $C_1, \ldots, C_n$ is called an* R-deduction *of C from $\mathcal{C}$ if it fulfills the following conditions: $C_n \equiv C$ and for all $i = 1, \ldots, n$:*

- *$C_i \in \mathcal{C}$ or*
- *C_i is an instance or a p-reduct of C_j for some $j < i$ or*
- *C_i is a resolvent of $\{C_j, C_k\}$ for some $j, k < i$ or*
- *C_i is the result of paramodulation of $\{C_j, C_k\}$ for $j, k < i$.*

An R-deduction of the empty sequent $\vdash$ from $\mathcal{C}$ is called an R-refutation of $\mathcal{C}$.

Finally, we state some lemmas that show that R-deductions can be transformed to **LKDe²**-proofs. These will be useful for showing the effectiveness of the CERES^2 method in the next section.

Lemma 1. *Let $C \equiv \Gamma \vdash \Delta$ be a clause, $\mathcal{D}$ be a set of clauses, ψ be a* **LKDe²***-proof of $\Gamma, \Pi \vdash \Lambda, \Delta$ from $\mathcal{D}$ with only quantifier-free cuts, let σ be a quantifier-free substitution whose domain contains no variable which occurs free in $\Pi \cup \Lambda$ and let $\Gamma^* \vdash \Delta^* \in \mathcal{S}(\{C\}, \sigma)$. Then we can construct an* **LKDe²***-proof ψ^* of $\Gamma^*, \Pi \vdash \Lambda, \Delta^*$ from $\mathcal{S}(\mathcal{D}, \sigma)$ with only quantifier-free cuts and with $|\psi^*| \leq |\psi| + \rho(|\Gamma\sigma \vdash \Delta\sigma|)$, where ρ is exponential if σ substitutes for a predicate variable in $\mathcal{D}$, and polynomial otherwise.*

Proof. By simulating the conjunctive normal form transformation in **LKDe²** using cuts. For a complete proof, see [6].

Note that this result is weaker than the corresponding result in first-order logic: the proofs constructed in that setting do not contain any cuts. Still, as our interest is the extraction of information, this result suffices, as quantifier-free cuts do not contain interesting mathematical information.

Lemma 2. *Let R be an R-deduction of $\Gamma \vdash \Delta$ from a set of clauses $\mathcal{C}$. Then there exists an* **LKDe²***-proof ψ of $\Gamma \vdash \Delta$ from $\mathcal{D}$ containing quantifier-free cuts only, where $\mathcal{D} = \{D \mid D \in \mathcal{S}(\mathcal{C}, \sigma)$ for some quantifier-free $\sigma\}$.*

Proof. By using Lemma 1 to replace instantiations, replacing resolution with cut, replacing paramodulation by the equality rules, and replacing p-reducts by contractions. For a full proof, see [6].

4 The CERES^2 cut-elimination method

We now define the CERES^2 method, which will turn out to be a cut-elimination method for **LKDe^2**-proofs using quantifier-free comprehension.

Definition 7. *Let (R) be a weak second-order quantifier rule*

$$\frac{A\{X \leftarrow \lambda x.F\}, \Gamma \vdash \Delta}{(\forall X)A, \Gamma \vdash \Delta}\ \forall^2 : l \qquad \frac{\Gamma \vdash \Delta, A\{X \leftarrow \lambda x.F\}}{\Gamma \vdash \Delta, (\exists X)A}\ \exists^2 : r$$

then (R) is called quantifier-free if F does not contain quantifiers. We call an **LKDe^2***-proof π a* **QFC***-proof if all its weak second-order quantifier rule applications are quantifier-free.*

Note that as we allow non-tautological axioms, it is not in general possible to eliminate all cuts. This leads to the following notion: An **LKDe^2**-proof π is called in *atomic cut normal form (ACNF)* if all cut-formulas of π are atomic.

An important technical tool in the CERES^2 method is the skolemization of proofs: this transformation removes strong quantifier rules from proofs and replaces the respective variables by Skolem terms.

Definition 8. *Let ψ be an* **LKDe^2***-proof. If the active formulas of all strong quantifier rules in ψ are ancestors of cut-formulas, then ψ is said to be in* Skolem form.

The following proposition shows that from a **QFC**-proof, we can indeed obtain a proof in Skolem form. Proofs in Skolem form allow the definition of proof projections by leaving out rules from the proof, as no eigenvariable violations can occur by doing so. This will be necessary to construct sound proofs in Definition 10. We use the structural skolemization operator sk on formulas and sequents, where sk replaces the strongly quantified variables by Skolem terms and drops the corresponding quantifiers (see [12]).

Proposition 1. *For every* **QFC***-proof ψ of S there exists a* **QFC***-proof ψ' of* $\mathrm{sk}(S)$ *in Skolem form.*

Proof. We obtain ψ' from ψ by dropping the strong quantifier rules going into the end-sequent and, on the path to the end-sequent, replacing the strongly quantified variables by the respective Skolem terms. For example, if S contains a positive occurrence α of $(\forall X)A(X)$ and the premise of the $\forall^2 : r$ introducing this quantifier is $\Pi \vdash \Lambda, A'(\Theta)$ and $t_1, \ldots, t_n$ are the (lambda-)terms eliminated by introductions of weak quantifiers dominating α and the corresponding skolem term in $\mathrm{sk}(S)$ is $\lambda z.P(z, x_1, \ldots, x_n)$, then we remove the $\forall^2 : r$ rule, replace its premise by $\Pi \vdash \Lambda, A'(\lambda z.P(z, t_1, \ldots, t_n))$, and modify the path to the end-sequent so that the occurrences of $t_1, \ldots, t_n$ in the Skolem term are eliminated by the weak quantifier rules. For a full proof, see [6].

Note that in this context, skolemization indeed does preserve validity (in contrast to what is observed in [13]), because the proposition we just stated generates a

proof of the skolemized formula from a proof of the unskolemized formula. As $\mathbf{LKDe^2}$ is sound, the transformation is validity preserving. We now define some notation that will be useful in describing CERES^2.

Definition 9. *Let ρ be a unary rule, σ a binary rule, ψ, χ* **QFC***-proofs, then $\rho(\psi)$ is the* **QFC***-proof obtained by applying ρ to the end-sequent of ψ, and $\sigma(\psi, \chi)$ is the proof obtained from the proofs ψ and χ by applying σ.*

Let P, Q be sets of **QFC***-proofs. Then $P^{\Gamma \vdash \Delta} := \{\psi^{\Gamma \vdash \Delta} \mid \psi \in P\}$, where $\psi^{\Gamma \vdash \Delta}$ is ψ followed by weakenings adding $\Gamma \vdash \Delta$, and $P \times_\sigma Q := \{\sigma(\psi, \chi) \mid \psi \in P, \chi \in Q\}$.*

Let $C = \{\Gamma_1 \vdash \Delta_1, \ldots, \Gamma_m \vdash \Delta_m\}$, $D = \{\Pi_1 \vdash \Lambda_1, \ldots, \Pi_n \vdash \Lambda_n\}$ be sets of clauses, then $C \times D := \{\Gamma_i, \Pi_j \vdash \Delta_i, \Lambda_j \mid i \leq m, j \leq n\}$.

We can now define the main parts of the CERES^2-method: the characteristic clause set and the set of proof projections of a proof π. The former will be always unsatisfiable and give rise to a resolution refutation, while the latter will allow the resolution refutation to be transformed into a proof of the end-sequent of π.

Definition 10. *Let π be a* **QFC***-proof in Skolem form. For each rule ρ in π, we define a set of cut-free* **QFC***-proofs, the set of* projections $\mathcal{P}_\rho(\pi)$ *of π, and a set of clauses, the* characteristic clause set $\mathrm{CL}_\rho(\pi)$ *of π, at the position of ρ.*

- *If ρ corresponds to an initial sequent, let $\Gamma_1 \vdash \Delta_1$ be the part of it which consists of ancestors of cut formulas, let $\Gamma_2 \vdash \Delta_2$ be the part which consists of ancestors of the end-sequent of π and define*
$$\begin{aligned} \mathcal{P}_\rho(\pi) &:= \{\Gamma_1, \Gamma_2 \vdash \Delta_2, \Delta_1\} \\ \mathrm{CL}_\rho(\pi) &:= \{\Gamma_1 \vdash \Delta_1\}. \end{aligned}$$
- *If ρ is a unary rule with immediate predecessor ρ' with $\mathcal{P}_{\rho'}(\pi) = \{\psi_1, \ldots, \psi_n\}$, distinguish:*
  (a) The active formulas of ρ are ancestors of cut formulas. Then
  $$\mathcal{P}_\rho(\pi) := \mathcal{P}_{\rho'}(\pi)$$
  (b) The active formulas of ρ are ancestors of the end-sequent. Then
  $$\mathcal{P}_\rho(\pi) := \{\rho(\psi_1), \ldots, \rho(\psi_n)\}$$
  Note that by assumption, all strong quantifier rules go into cuts, so ρ cannot be a strong quantifier rule, so no eigenvariable violation can occur here.

  In any case, $\mathrm{CL}_\rho(\pi) := \mathrm{CL}_{\rho'}(\pi)$.
- *Let ρ be a binary rule with immediate predecessors ρ_1 and ρ_2.*
  (a) If the active formulas of ρ are ancestors of cut-formulas, let $\Gamma_i \vdash \Delta_i$ be the ancestors of the end-sequent in the conclusion sequent of ρ_i and define
  $$\mathcal{P}_\rho(\pi) := \mathcal{P}_{\rho_1}(\pi)^{\Gamma_2 \vdash \Delta_2} \cup \mathcal{P}_{\rho_2}(\pi)^{\Gamma_1 \vdash \Delta_1}$$
  For the characteristic clause set, define
  $$\mathrm{CL}_\rho(\pi) := \mathrm{CL}_{\rho_1}(\pi) \cup \mathrm{CL}_{\rho_2}(\pi)$$

(b) If the active formulas of ρ are ancestors of the end-sequent, then

$$\mathcal{P}_\rho(\pi) := \mathcal{P}_{\rho_1}(\pi) \times_\rho \mathcal{P}_{\rho_2}(\pi).$$

For the characteristic clause set, define

$$\mathrm{CL}_\rho(\pi) := \mathrm{CL}_{\rho_1}(\pi) \times \mathrm{CL}_{\rho_2}(\pi)$$

The set of projections *of π, $\mathcal{P}(\pi)$ is defined as $\mathcal{P}_{\rho_0}(\pi)$, and the* characteristic clause set *of π, $\mathrm{CL}(\pi)$ is defined as $\mathrm{CL}_{\rho_0}(\pi)$, where ρ_0 is the last rule of π.*

Example 1. Consider the proof ψ:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{a \in \Theta \vdash a \in \Theta}{\vdash a \in \Theta, a \notin \Theta}\neg : r \quad \dfrac{b \in \Theta \vdash b \in \Theta}{b \notin \Theta, b \in \Theta \vdash}\neg : l}{b \in \Theta, a \notin \Theta \to b \notin \Theta \vdash a \in \Theta}\to : l}{b \in \Theta, (\forall X)(a \in X \to b \in X) \vdash a \in \Theta}\forall^2 : l}{(\forall X)(a \in X \to b \in X) \vdash b \in \Theta \to a \in \Theta}\to : r}{(\forall X)(a \in X \to b \in X) \vdash (\forall X)(b \in X \to a \in X)}\forall^2 : r \quad \dfrac{\dfrac{\dfrac{b \in P \vdash b \in P \qquad a \in P \vdash a \in P}{b \in P \to a \in P, b \in P \vdash a \in P}\to : l}{b \in P \to a \in P \vdash b \in P \to a \in P}\to : r}{(\forall X)(b \in X \to a \in X) \vdash b \in P \to a \in P}\forall^2 : l}{(\forall X)(a \in X \to b \in X) \vdash b \in P \to a \in P}\,cut$$

where X, Θ are predicate variables, a, b are individual constants, P is a predicate constant, and the lambda terms used in the $\forall^2 : l$ rules are $\lambda x.x \notin \Theta$ and $\lambda x.x \in P$. Then

$$\begin{aligned}\mathrm{CL}(\psi) &= (\{\vdash a \in \Theta\} \times \{b \in \Theta \vdash\}) \cup \{\vdash b \in P\} \cup \{a \in P \vdash\} \\ &= \{b \in \Theta \vdash a \in \Theta;\ \vdash b \in P;\ \ a \in P \vdash\}\end{aligned}$$

and $\mathcal{P}(\psi)$ contains, among others, the proof

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{a \in \Theta \vdash a \in \Theta}{\vdash a \in \Theta, a \notin \Theta}\neg : r \quad \dfrac{b \in \Theta \vdash b \in \Theta}{b \notin \Theta, b \in \Theta \vdash}\neg : l}{b \in \Theta, a \notin \Theta \to b \notin \Theta \vdash a \in \Theta}\to : l}{b \in \Theta, (\forall X)(a \in X \to b \in X) \vdash a \in \Theta}\forall^2 : l}{b \in \Theta, (\forall X)(a \in X \to b \in X) \vdash b \in P \to a \in P, a \in \Theta}\,w : r$$

Note that for the soundness of Definition 10, we need the assumption that π is in Skolem form: if this were not the case, violations of eigenvariable conditions could appear in the projections.

We will now prove the main properties of CERES^2.

Lemma 3. *Let π be a* **QFC***-proof in Skolem form. Then there exists an R-refutation of* $\mathrm{CL}(\pi)$*.*

Proof. Analogous to the proof of unsatisfiability of $\mathrm{CL}(\pi)$ for first-order logic in [3] by removing all rules of π except the ancestors of the cuts, and removing all formula occurrences in π except the ancestors of cuts, we construct a **QFC**-proof ψ of $\vdash$ from $\mathrm{CL}(\pi)$. By applying reductive cut-elimination (see e.g. [14]) to ψ, we obtain a **QFC**-proof ψ' of $\vdash$ using atomic cut, contraction and permutation only. We find that the initial sequents of ψ' are all instances of clauses in $\mathrm{CL}(\pi)$. It is then easy to convert ψ' into an R-refutation. For a full proof, see [6].

Note that this lemma is just a theoretical tool to show the existence of a suitable refutation — in practice, the reductive methods used in the proof of the lemma are not used (as can be seen in the analysis of the example in Section 5).

We are now ready to define the CERES^2 method and state our central result.

Definition 11. *Let π be a* **QFC***-proof of S. Then the* CERES^2 *method is the following algorithm:*

1. *Compute a* **QFC***-proof π_{sk} of* $\mathrm{sk}(S)$.
2. *Compute* $\mathrm{CL}(\pi_{sk})$, $\mathcal{P}(\pi_{sk})$.
3. *Compute an R-refutation γ of* $\mathrm{CL}(\pi_{sk})$.
4. *Convert γ into an* **LKDe^2***-proof γ' of $\vdash$ from* $\mathrm{CL}(\pi_{sk})$.
5. *Plug instances of the proofs in $\mathcal{P}(\pi_{sk})$ into the leaves of γ' to obtain a proof ψ of* $\mathrm{sk}(S)$ *containing quantifier-free cuts only.*
6. *Perform quantifier-free cut-elimination on ψ to obtain a proof φ of* $\mathrm{sk}(S)$ *containing no non-atomic cuts.*

Let us remark here that in step 6, any method for cut-elimination for quantifier-free cuts can be used (e.g. reductive methods, "zero-th order" CERES). Furthermore, considering that the instantiations of quantifiers are the core information in a proof, one can even leave out this step as the instantiations in φ and ψ coincide.

Theorem 1. *Let π be a* **QFC***-proof of S. Then the* CERES^2 *method transforms π into an* **LKDe^2***-proof φ of* $\mathrm{sk}(S)$ *such that φ is in atomic-cut normal form.*

Proof. Using Proposition 1, we convert π to π_{sk}. By Lemma 3, there exists an R-refutation γ of $\mathrm{CL}(\pi_{sk})$. By Lemma 2, from γ we can construct an **LKDe^2**-refutation γ' of $\mathrm{CL}(\pi_{sk})$. Every initial sequent of γ' is either a sequent $A \vdash A$, an axiom, or an instance C^* of some $C \in \mathrm{CL}(\pi_{sk})$ under a substitution σ. Let $C \equiv \Pi \vdash \Lambda$ and $\mathrm{sk}(S) \equiv \Gamma \vdash \Delta$, then by Definition 10 we have a cut-free **QFC**-proof ψ_C of $\Gamma, \Pi \vdash \Lambda, \Delta$. Let $C^* \equiv \Pi^* \vdash \Lambda^*$, then by Lemma 1, we can construct **LKDe^2**-proofs ψ_{C^*} of $\Gamma, \Pi^* \vdash \Lambda^*, \Delta$ that contain quantifier-free cuts only. By plugging these proofs onto the leaves of γ' and adding contractions at the end, we obtain an **LKDe^2**-proof of $\Gamma \vdash \Delta$ containing quantifier-free cuts only. By applying cut-elimination to this proof, we obtain the desired proof φ.

4.1 Extending CERES^2

This work defines a method for cut-elimination for **QFC**-proofs. A natural question is then, whether the method can be extended to stronger comprehension. In the previous section, it was stated that skolemization is an important technical tool in the context of the method, as it removes strong quantifier introduction rules and because of this allows the definition of proof projections without causing violations of eigenvariable conditions.

When considering comprehension involving quantifiers, proof skolemization has to be modified to achieve the same effect: it is not enough to skolemize the end-sequent, as the active formulas of strong quantifier rules may be ancestors

of formulas removed by weak second-order quantifier rules and therefore, the corresponding strong quantifiers will not be present in the end-sequent.

A tempting idea is, then, to simply skolemize the formulas that disappear into weak second-order quantifier rules. This approach is investigated in [6] and it turns out that weak quantifier rules cannot be skolemized within **LKDe²** in most cases; the class where this is possible is only slightly larger than **QFC** and looks rather unnatural. So either we have to extend proof skolemization to more involved proof transformations or new techniques for dealing with projections containing strong quantifier rules have to be developed. A promising approach is to use strong quantifier rules which introduce a quantifier not from a free variable but from a Skolem term as in [13].

5 CERES² Example

We will now apply the CERES^2 method to a **QFC**-proof φ. The proof under consideration is a proof of the theorem $\sum_{i=0}^{n} i = \frac{n(n+1)}{2}$ by the least number principle. As axioms the proof uses elementary axioms of arithmetic such as associativity and commutativity of $+$ and $*$, axioms for the neutral elements 0 and 1, and distributivity. The following axioms represent the recursive definition of the series:

$$\vdash \Sigma(n+1) = \Sigma(n) + (n+1) \; ; \vdash \Sigma(0) = 0$$

We write 2 for $1+1$. In the proof, $\star$ denotes the ancestors of a cut, double lines indicate applications of propositional rules, and structural rules except cut are omitted.

$\varphi :=$

$$\dfrac{\begin{array}{cc} \varphi_1 & \varphi_2 \\ \vdots & \vdots \\ LNP \vdash IND^\star & IND^\star \vdash (\forall n) 2 * \Sigma(n) = n * (n+1) \end{array}}{LNP \vdash (\forall n) 2 * \Sigma(n) = n * (n+1)} \; cut$$

where

$$\begin{aligned} LNP &\equiv (\forall Y)((\exists z) z \in Y \rightarrow 0 \in Y \vee (\exists z)(z \notin Y \wedge z+1 \in Y)) \\ IND &\equiv (\forall X)(0 \in X \wedge (\forall y)(y \in X \rightarrow y+1 \in X) \rightarrow (\forall y) y \in X) \end{aligned}$$

This proof uses the fact that the least number principle implies induction as a lemma; the use of this lemma will be removed by application of the CERES^2 method, yielding a new proof that shows that the least number principle implies the theorem, without the use of induction.

The proof φ_1 specified below is exactly the proof of this lemma, and it is a formalization of the following argument: Assume the least number principle, and assume that for an arbitrary set $\mathcal{X}$, $0 \in \mathcal{X}$ and if $y \in \mathcal{X}$, then $y+1 \in \mathcal{X}$, and assume for contradiction that $\mathcal{X} \neq \mathbb{N}$. Then the set $\bar{\mathcal{X}} = \{x \mid x \notin \mathcal{X}\}$ (or $\lambda x.x \notin X$ in the lambda notation) is not empty, so by the least number principle either

1. $0 \in \bar{\mathcal{X}}$. But $0 \in \mathcal{X}$ by assumption, so $0 \notin \bar{\mathcal{X}}$.
2. There is a z s.t. $z \notin \bar{\mathcal{X}}$ and $z+1 \in \bar{\mathcal{X}}$. But then $z \in \mathcal{X}$ and by assumption $z+1 \in \mathcal{X}$, so $z+1 \notin \bar{\mathcal{X}}$.

So φ_1 is

$$\dfrac{\dfrac{\dfrac{\dfrac{y_0 \in X_0 \vdash y_0 \in X_0^\star}{\vdash (\forall y) y \in X_0^\star, (\exists z) z \notin X_0}\,\forall : r, \exists : r \qquad \varphi_1^1}{0 \in X_0^\star, (\forall y)(y \in X_0 \to y+1 \in X_0)^\star, LNP_\sigma \vdash (\forall y) y \in X_0^\star}\,\to : l}{LNP \vdash 0 \in X_0 \wedge (\forall y)(y \in X_0 \to y+1 \in X_0) \to (\forall y) y \in X_0^\star}\,\forall^2 : l\ \lambda x.x \notin X_0}{LNP \vdash IND^\star}\,\forall^2 : r$$

where

$$LNP_\sigma \equiv (\exists z) z \notin X_0 \to 0 \notin X_0 \vee (\exists z)(\neg z \notin X_0 \wedge z+1 \notin X_0)$$

The proof φ_1^1 is

$$\dfrac{\dfrac{0 \in X_0^\star \vdash 0 \in X_0}{0 \in X_0^\star, 0 \notin X_0 \vdash}\,\neg : l \qquad \dfrac{\dfrac{z_0 \in X_0 \vdash z_0 \in X_0^\star \qquad z_0+1 \in X_0^\star \vdash z_0+1 \in X_0}{z_0 \in X_0 \to z_0+1 \in X_0^\star, \neg z_0 \notin X_0 \wedge z_0+1 \notin X_0 \vdash}}{(\forall y)(y \in X_0 \to y+1 \in X_0)^\star, (\exists z)(\neg z \notin X_0 \wedge z+1 \notin X_0) \vdash}\,\exists, \forall : l}{0 \in X_0^\star, (\forall y)(y \in X_0 \to y+1 \in X_0)^\star, 0 \notin X_0 \vee (\exists z)(\neg z \notin X_0 \wedge z+1 \notin X_0) \vdash}\,\vee : l$$

This completes the left hand side of the cut, showing that the least number principle implies induction. The right hand side of the cut is a formalization of the following induction proof of $\sum_{i=0}^{n} i = \frac{n(n+1)}{2}$: The induction base is trivial. For the induction step we want to show

$$\sum_{i=0}^{n+1} i = n+1+\sum_{i=0}^{n} i = \frac{(n+1)((n+1)+1)}{2}$$

By the induction hypothesis this reduces to showing

$$n+1+\frac{n(n+1)}{2} = \frac{(n+1)((n+1)+1)}{2}$$

which clearly holds.

The formalization of this argument is the proof φ_2:

$$\dfrac{\varphi_2^1 \qquad \dfrac{\dfrac{2 * \Sigma(n_0) = n_0 * (n_0+1)^\star \vdash 2 * \Sigma(n_0) = n_0 * (n_0+1)}{(\forall x) 2 * \Sigma(x) = x * (x+1)^\star \vdash (\forall n) 2 * \Sigma(n) = n * (n+1)}\,\forall : r, \forall : l}{IND_\sigma^\star \vdash (\forall n) 2 * \Sigma(n) = n * (n+1)}\,\to : l}{IND^\star \vdash (\forall n) 2 * \Sigma(n) = n * (n+1)}\,\forall^2 : l\ \lambda x.2 * \Sigma(x) = x * (x+1)$$

where

$$IND_\sigma \equiv 2 * \Sigma(0) = 0 * (0+1) \wedge (\forall x)(2 * \Sigma(x) = x * (x+1) \to 2 * \Sigma(x+1) = (x+1) * ((x+1)+1)) \to (\forall x) 2 * \Sigma(x) = x * (x+1)$$

We continue with φ_2^1 — from this point on, we will omit $\star$ as all formula occurrences in the following proofs are cut ancestors:

$$\dfrac{\dfrac{\vdash \Sigma(0) = 0 \qquad \dfrac{\vdash 2*0 = 0 \qquad \vdash 0 = 0*(0+1)}{\vdash 2*0 = 0*(0+1)} =: r_2}{\vdash 2*\Sigma(0) = 0*(0+1)} =: r_2 \qquad \varphi_2^2}{\vdash 2*\Sigma(0) = 0*(0+1) \wedge (\forall x)(2*\Sigma(x) = x*(x+1) \rightarrow 2*\Sigma(x+1) = (x+1)*((x+1)+1))} \wedge$$

Note that the left branch of φ_2^1 proves the induction base. The proof φ_2^2 will in turn show the induction step:

$$\dfrac{\dfrac{\vdash \Sigma(x_0+1) = \Sigma(x_0) + (x_0+1) \qquad \varphi_1^=}{2*\Sigma(x_0) = x_0*(x_0+1) \vdash 2*\Sigma(x_0+1) = (x_0+1)*((x_0+1)+1)} =: r_2}{\vdash (\forall x)(2*\Sigma(x) = x*(x+1) \rightarrow 2*\Sigma(x+1) = (x+1)*((x+1)+1))} \forall : r$$

where $\varphi_1^=$ is a proof of

$$2*\Sigma(x_0) = x_0*(x_0+1) \vdash 2*(\Sigma(x_0) + (x_0+1)) = (x_0+1)*((x_0+1)+1)$$

using purely equational reasoning. This completes the proof φ.

Skolemization of φ (for details on proof skolemization, refer to the proof of Proposition 1 in [6]) yields a proof φ_{sk} of the sequent

$$(\forall Y)((\exists z) z \in Y \rightarrow 0 \in Y \vee (f(Y) \notin Y \wedge f(Y) + 1 \in Y)) \vdash 2*\Sigma(s) = s*(s+1)$$

where f, s are the Skolem symbols. In the proof, the Skolem term $f(\lambda x.x \notin X_0)$ replaces the eigenvariable z_0 and the Skolem term s replaces the eigenvariable n_0.

Remark 1. In all models of arithmetic and the left hand side of the sequent, a suitable interpretation of f will be a function $\gamma : P(\mathbb{N}) \mapsto \mathbb{N}$ such that for all $S \in P(\mathbb{N})$ with $S \neq \emptyset, 0 \notin S$, we have $\gamma(S) = \min(S) - 1$. This is an example for the natural interpretation of Skolem symbols, which in practice is often possible.

The characteristic clause set $\mathrm{CL}(\varphi_{sk})$ can be written as:

$$\mathrm{CL}(\varphi_{sk}) = \mathrm{CL}(\varphi_{sk}^1) \cup \mathrm{CL}(\varphi_{sk}^2)$$

$$\begin{aligned}\mathrm{CL}(\varphi_{sk}^1) = (&\{0 \in X_0 \vdash\} \times \{\vdash f(\lambda x.x \notin X_0) \in X_0; f(\lambda x.x \notin X_0) + 1 \in X_0 \vdash\}) \\ &\times \{\vdash y_0 \in X_0\}\end{aligned}$$

$$\begin{aligned}\mathrm{CL}(\varphi_{sk}^2) = &\{2*\Sigma(s) = s*(s+1) \vdash\} \cup \{\vdash \Sigma(x_0+1) = \Sigma(x_0) + (x_0+1)\} \\ &\cup \{2*\Sigma(x_0) = x_0*(x_0+1) \vdash 2*\Sigma(x_0) = x_0*(x_0+1)\} \\ &\cup \{\vdash \Sigma(0) = 0\} \cup PAX_S\end{aligned}$$

where PAX_S is the set of axioms of arithmetic that are used in the proof φ_2^1. Modulo subsumption and tautology deletion, the characteristic clause set is:

$$\begin{aligned}CL(\varphi_{sk}) = \{\ & 0 \in X_0 \vdash f(\lambda x.x \notin X_0) \in X_0, y_0 \in X_0; && (I1)\\ & 0 \in X_0, f(\lambda x.x \notin X_0) + 1 \in X_0 \vdash y_0 \in X_0; && (I2)\\ & 2*\Sigma(s) = s*(s+1) \vdash; && (T1)\\ & \vdash \Sigma(x_0+1) = \Sigma(x_0) + (x_0+1); && (S1)\\ & \vdash \Sigma(0) = 0\} && (S2)\\ & \cup PAX'_S\end{aligned}$$

where PAX'_S is PAX_S after subsumption and tautology deletion.

5.1 Refutation of the characteristic clause set

We now define a resolution refutation of the characteristic clause set $CL(\varphi_{sk})$, using the resolution calculus from Section 3.

The clauses $(I1)$ and $(I2)$ correspond to the induction axiom, while the clause $(T1)$ is the negated theorem. For the refutation we will need the following instances of the induction clauses produced from the substitution $\sigma = \langle\{y_0 \leftarrow s\}, \{X_0 \leftarrow \lambda x.2 * \Sigma(x) = x * (x + 1)\}\rangle$:

$$(I1')\ 2 * \Sigma(0) = 0 * (0 + 1) \vdash 2 * \Sigma(f(T)) = f(T) * (f(T) + 1), 2 * \Sigma(s) = s * (s + 1)$$

$$(I2')\ 2 * \Sigma(0) = 0 * (0 + 1), 2 * \Sigma(f(T) + 1) = (f(T) + 1) * ((f(T) + 1) + 1) \vdash 2 * \Sigma(s) = s * (s + 1)$$

where $T \equiv \lambda x.\neg 2 * \Sigma(x) = x * (x + 1)$. We start by deriving the induction base using resolution, for this we need the clauses

$$(A1) \vdash 2 * 0 = 0\ ;\ (A2) \vdash 0 = 0 * (0 + 1)$$

Note that $(A1), (A2) \in PAX_S$. We now use paramodulation from $(S2)$ into $(A1)$ to derive

$$(IB1) \vdash 2 * \Sigma(0) = 0$$

Paramodulation from $(IB1)$ into $(A2)$ then yields

$$(IB) \vdash 2 * \Sigma(0) = 0 * (0 + 1)$$

We now resolve both $(I1')$ and $(I2')$ first with (IB) and then with $(T1)$ to obtain

$$(IH) \vdash 2 * \Sigma(f(T)) = f(T) * (f(T) + 1)$$
$$(IG)\ 2 * \Sigma(f(T) + 1) = (f(T) + 1) * ((f(T) + 1) + 1) \vdash$$

Note that (IH) corresponds to the induction hypothesis in the original proof, while (IG) is the negation of what was proved in the induction step. Towards a contradiction, we paramodulate (IG) with an instance of the second part of the definition of the series, $(S1)$, and get

$$(C1)\ 2 * (\Sigma(f(T)) + (f(T) + 1)) = (f(T) + 1) * ((f(T) + 1) + 1) \vdash$$

From clauses from PAX_S, it is easy to derive (using paramodulation exclusively) the clause

$$(C2) \vdash 2 * (\Sigma(x_0) + (x_0 + 1)) = 2 * \Sigma(x_0) + 2 * (x_0 + 1)$$

Paramodulation from an instance of $(C2)$ into $(C1)$ yields

$$(C3)\ 2 * \Sigma(f(T)) + 2 * (f(T) + 1) = (f(T) + 1) * ((f(T) + 1) + 1) \vdash$$

We can now use paramodulation to obtain from $(C3)$ and (IH) the clause

$$(C4)\ (f(T) * (f(T) + 1)) + 2 * (f(T) + 1) = (f(T) + 1) * ((f(T) + 1) + 1) \vdash$$

which is a wrong arithmetical statement. From clauses in PAX_S it is now easy to derive the dual clause (modulo substitution)

$$(C5) \vdash x_0 * (x_0 + 1) + 2 * (x_0 + 1) = (x_0 + 1) * ((x_0 + 1) + 1)$$

We can now resolve $(C4)$ with an instance of $(C5)$ to obtain the empty sequent and complete the refutation. Note that although the clauses used in the refutation correspond to the induction axiom, the proof constructed from the refutation will be a proof by the least number principle. This will become clear in the next section.

5.2 Interpretation of the ACNF

In this section we will indicate the construction of the ACNF from the refutation of $CL(\varphi_{sk})$ produced in the previous section. We will not give the full ACNF in this section, as it is too large to display comfortably, but we will discuss its key features.

The key information of a cut-free proof lies in the instantiations of the quantifiers (the other information just pertains to propositional reasoning and structural manipulation of sequents), so we will investigate a projection that contains such instantiations, namely the projection $\varphi[(I1)]$ to the clause

$$(I1) \equiv 0 \in X_0 \vdash f(S) \in X_0, y_0 \in X_0$$

where $S \equiv \lambda x.x \notin X_0$:

$$\dfrac{\dfrac{\dfrac{\dfrac{y_0 \in X_0 \vdash y_0 \in X_0}{\vdash y_0 \notin X_0, y_0 \in X_0}\neg : r}{\vdash (\exists z)z \notin X_0, y_0 \in X_0}\exists : r \qquad \psi}{0 \in X_0, (\exists z)z \notin X_0 \rightarrow (0 \notin X_0 \vee (\neg f(S) \notin X_0 \wedge f(S) + 1 \notin X_0)) \vdash \Delta}\rightarrow : l}{0 \in X_0, (\forall Y)((\exists z)z \in Y \rightarrow (0 \in Y \vee (f(Y) \notin Y \wedge f(Y) + 1 \in Y))) \vdash \Delta}\forall^2 : l\ \lambda x.x \notin X_0$$

where $\Delta \equiv f(S) \in X_0, y_0 \in X_0$ and ψ only consists of propositional inferences from tautological initial sequents. In the refutation, the instance of $(I1)$ under the substitution $\sigma = \langle \{y_0 \leftarrow s\}, \{X_0 \leftarrow \lambda x.2 * \Sigma(x) = x * (x + 1)\}\rangle$ is used, therefore the projection used in the construction of the ACNF is $\varphi[(I1)]\sigma$ (cf. Lemma 1). This yields a projection with a rule application $\forall^2 : l\ \lambda x.\neg(2 * \Sigma(x) = x * (x + 1))$. This use of comprehension is the key point in the argument of the ACNF: while the proof by induction showed that the formula holds for all n (or in other words, all n are in the set X), the proof by the least number property shows that the negation of the formula holds for no n (or that $\bar{X}$ is empty).

It is interesting to note that no matter what theorem is proved by induction in the input proof, the proof of $LNP \vdash IND$ remains the same and therefore also $(I1)$ and $\varphi[(I1)]$ remain unchanged. So as long as the clause $(I1)$ is used in the resolution refutation, the resulting ACNF will contain the above argument where only the definition of the set X differs.

6 Future Work

There is still much to be done: We are working on extending CERES^2 to larger classes of proofs and investigating the use of existing higher-order resolution calculi (see e.g. [15]) with CERES^2. For semi-automated application of the method, it will be necessary to replace the unrestricted substitution of our resolution calculus by unification (see e.g. [16]). Finally, the existing ANSI C++ implementation of CERES is being extended to CERES^2. This will allow practical application of the method to larger and more interesting proofs.

References

1. Kohlenbach, U.: Effective bounds from ineffective proofs in analysis: an application of functional interpretation and majorization. Journal of Symbolic Logic **57**(4) (1992) 1239–1273
2. Baaz, M., Leitsch, A.: Towards a clausal analysis of cut-elimination. Journal of Symbolic Computation **41** (2006) 381–410
3. Baaz, M., Leitsch, A.: Cut-elimination and Redundancy-elimination by Resolution. Journal of Symbolic Computation **29**(2) (2000) 149–176
4. Baaz, M., Hetzl, S., Leitsch, A., Richter, C., Spohr, H.: Ceres: An Analysis of Fürstenberg's Proof of the Infinity of Primes. Theoretical Computer Science **403** (2008) 160–175
5. Hetzl, S.: Characteristic Clause Sets and Proof Transformations. PhD thesis, Vienna University of Technology (2007)
6. Hetzl, S., Leitsch, A., Weller, D., Woltzenlogel Paleo, B.: CERES in second-order logic. Technical report, Vienna University of Technology (2008) available at `http://www.logic.at/ceres/downloads/docs/report_ceres2.pdf`.
7. Church, A.: A formulation of the simple theory of types. Journal of Symbolic Logic **5**(2) (1940) 56–68
8. Boolos, G.S., Burgess, J.P., Jeffrey, R.C.: Computability and Logic. 4th edn. Cambridge University Press, Cambridge, UK (2002)
9. Takeuti, G.: Proof Theory. Volume 81 of Studies in Logic and the Foundations of Mathematics. North-Holland Publishing Co. (1975)
10. Baaz, M., Hetzl, S., Leitsch, A., Richter, C., Spohr, H.: Proof transformation by CERES. In: Lecture Notes in Artificial Intelligence. Volume 4108., Mathematical Knowledge Management, Springer Berlin (2006) 82–93
11. Andrews, P.B.: Resolution in Type Theory. Journal of Symbolic Logic **36**(3) (1971) 414–432
12. Baaz, M., Leitsch, A.: Cut normal forms and proof complexity. Annals of Pure and Applied Logic **97** (1999) 127–177
13. Miller, D.A.: A compact representation of proofs. Studia Logica **46**(4) (1987) 347–370
14. Danos, V., Joinet, J.B., Schellinx, H.: A New Deconstructive Logic: Linear Logic. Journal of Symbolic Logic **62**(3) (1997) 755–807
15. Benzmüller, C.: Comparing approaches to resolution based higher-order theorem proving. Synthese **133**(1–2) (2002) 203–335
16. Dowek, G.: Higher-order unification and matching. In: Handbook of automated reasoning. Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands (2001) 1009–1062