CHRISTOPH BENZMÜLLER

COMPARING APPROACHES TO RESOLUTION BASED HIGHER-ORDER THEOREM PROVING

ABSTRACT. We investigate several approaches to resolution based automated theorem proving in classical higher-order logic (based on Church's simply typed λ-calculus) and discuss their requirements with respect to Henkin completeness and full extensionality. In particular we focus on Andrews' higher-order resolution (Andrews 1971), Huet's constrained resolution (Huet 1972), higher-order E-resolution, and extensional higher-order resolution (Benzmüller and Kohlhase 1997). With the help of examples we illustrate the parallels and differences of the extensionality treatment of these approaches and demonstrate that extensional higher-order resolution is the sole approach that can completely avoid additional extensionality axioms.

1. INTRODUCTION

It is a well known consequence of Gödel's first incompleteness theorem that there cannot be complete calculi for higher-order logic with respect to standard semantics. However, Henkin (1950) showed that there are indeed complete calculi if one gives up the intuitive requirement of full function domains in standard semantics and considers Henkin's general models instead. For higher-order calculi therefore Henkin completeness constitutes the most interesting notion of completeness.

A very challenging task for a calculus aiming at Henkin-completeness is to provide a suitable extensionality treatment. Unfortunately the importance of full extensionality in higher-order theorem proving, i.e., the suitable combination of functional and Boolean extensionality, has widely been overlooked so far. This might be due to the fact that (weak) functional extensionality is already built-in in the pure simply typed λ-calculus and that Boolean extensionality or the subtle interplay between Boolean and functional extensionality does simply not occur in this context. However, the situation drastically changes as soon as one is interested in a higher-order logic based on the simply typed λ-calculus, as now Boolean extensionality is of importance too.

We therefore investigate the extensionality treatment of several resolution based approaches to Henkin complete higher-order theorem proving:

Synthese **133:** 203–235, 2002.

Andrews' higher-order resolution (Andrews 1971), Huet's constrained resolution (Huet 1972), higher-order E-resolution, and extensional higher-order resolution (Benzmüller and Kohlhase 1998a). In order to ease the comparison we present them in a uniform way. Even though we focus on the resolution method in this paper the main results on the feasibility of extensionality reasoning in higher-order theorem proving do nevertheless apply to other theorem proving approaches as well.

For Andrews' and Huet's approach it is well known that generally infinitely many extensionality axioms are required in the search space in order to reach Henkin completeness. With the help of rather simple examples we will point out the shortcomings of this kind of extensionality treatment; namely a fair amount of non-goal directed search which contrasts the general idea of resolution based theorem proving.

Whereas the use of higher-order E-unification (cf. Snyder 1990; Nipkow and Qian 1991; Wolfram 1993; Qian and Wang 1996) instead of simple syntactical higher-order unification partially improves the situation, this idea nevertheless fails to provide a general solution and still requires additional extensionality axioms to ensure Henkin completeness.

The first calculus that generally takes into account, that higher-order theory unification with respect to theories including full extensionality is as hard as Henkin complete higher-order theorem proving itself, is the extensional higher-order resolution approach (Benzmüller and Kohlhase 1998a). This calculus very closely integrates higher-order unification and resolution by allowing for mutual recursive calls (instead of hierarchical calls solely from resolution to unification as in first-order). With its close integration of unification and resolution this approach ensures Henkin completeness without requiring additional extensionality axioms. With the help of our examples we show that this aspect is not only of theoretical but also of practical importance as proof problems requiring non-trivial extensionality reasoning can be solved in the extensional higher-order resolution approach in a more goal directed way.

As a theoretical result the paper presents Henkin completeness proofs for the resolution approaches of Andrews and Huet which have been examined in literature so far only with respect to Andrews' rather weak semantical notion of V-complexes.

The paper is organised as follows: Syntax and semantics of higher-order logic and a proof theoretic tool for analysing Henkin completeness are sketched in Section 2. Various resolution based calculi are then introduced in Sections 3 and their extensionality treatment is investigated with the help of examples in Section 4. Related work is addressed in Section 5, and Section 6 concludes the paper.

2. SYNTAX AND SEMANTICS OF HIGHER-ORDER LOGIC

2.1. *Classical Type Theory*

We consider a higher-order logic based on Church's simply typed λ-calculus (Church 1940) and choose $BT := \{\iota, o\}$ as *base types*, where ι denotes the set of individuals and o the set of truth values. *Functional types* are inductively defined over BT. A *signature* Σ contains for each type an infinite set of variables and constants, and particularly it provides the *logical constants* $\neg_{o\to o}$, $\vee_{o\to o\to o}$, and $\Pi_{(\alpha\to o)\to o}$ for every type α. As all other logical operators can be defined (e.g., $\mathbf{A} \wedge \mathbf{B} := \neg(\neg\mathbf{A} \vee \neg\mathbf{B})$, $\forall X_\alpha.\, \mathbf{P}\ X := \Pi_{((\alpha\to o)\to o)}(\lambda X_\alpha.\, \mathbf{P}\ X)$, and $\exists X_\alpha.\, \mathbf{P}\ X := \neg\forall X_\alpha.\, \neg(\mathbf{P}\ X)$)) the given logical constants are sufficient to define a classical higher-order logic.

The set of all Σ-terms (closed Σ-terms) of type α is denoted by wff_α ($cwff_\alpha$). Variables are printed as upper-case (e.g., X_α), constants as lower-case letters (e.g., c_α), and arbitrary terms appear as bold capital letters (e.g., $\mathbf{T}_\alpha$). If the type of a symbol is uniquely determined by the given context we omit it. We abbreviate function applications by $h_{\alpha_1\to\cdots\to\alpha_n\to\beta}\ \overline{\mathbf{U}^n_{\alpha_n}}$, which stands for $(\cdots(h_{\alpha_1\to\cdots\to\alpha_n\to\beta}\ \mathbf{U}^1_{\alpha_1})\cdots\mathbf{U}^n_{\alpha_n})$. For α-, β-, η-, $\beta\eta$-*conversion* and the definition of β-*normal,* $\beta\eta$-*normal, long* $\beta\eta$-*normal*, and *head-normal form* we refer to Barendregt (1984) as well as for the definition of *free* variables, *closed* formulas (also called *sentences*), and *substitutions*. Substitutions are represented as $[\mathbf{T}_1/X_1, \ldots, \mathbf{T}_n/X_n]$ where the X_i specify the variables to be replaced by the terms $\mathbf{T}_i$. The application of a substitution σ to a term (resp. literal or clause) $\mathbf{C}$ is printed $\mathbf{C}_\sigma$.

Higher-order unification and sets of *partial bindings* $\mathcal{GB}^h_\gamma$ are well explained in Snyder and Gallier (1989).

A calculus R provides a set of rules $\{r_n \mid 0 < n \leq i\}$ defined on clauses. We write $\Phi \vdash^{r_n} \mathcal{C}$ ($\mathcal{C}' \vdash^{r_n} \mathcal{C}$) iff clause $\mathcal{C}$ is the result of a one step application of rule $r_n \in R$ to premise clauses $\mathcal{C}'_i \in \Phi$ (to $\mathcal{C}'$ respectively). Multiple step derivations in calculus R are abbreviated by $\Phi_1 \vdash_R \Phi_k$ (or $\mathcal{C}_1 \vdash_R \mathcal{C}_k$).

2.2. *Clauses, Literals, and Unification Constraints*

The approaches studied in this paper are presented using a uniform notation for clauses, literals, and unification constraints (the notation is due to Kohlhase (1994)). *Literals*, e.g., $[\mathbf{A}]^\mu$, consist of a *literal atom* $\mathbf{A}$ and a *polarity* $\mu \in \{T, F\}$. For all rules presented in this paper we assume that the polarity specifiers $\mu, \nu \in \{T, F\}$ refer to complementary polarities, i.e., $\mu \neq \nu$. In particular we distinguish between *proper literals* and *pre-literals*. The (normalised) atom of a pre-literal has a logical constant at

head position, whereas this must not be the case for proper literals. For instance, $[\mathbf{A} \vee \mathbf{B}]^T$ is a pre-literal and $[p_{o\to o}\ (\mathbf{A} \vee \mathbf{B})]^T$ is a proper literal. Furthermore a literal is called *flexible* if its atom contains a variable at head position.

A unification problem between two terms $\mathbf{T}^1$ and $\mathbf{T}^2$ (between n terms $\mathbf{T}^1, \ldots, \mathbf{T}^n$) generated during the refutation process is called an *unification constraint* and is represented as $[\mathbf{T}^1 \neq^? \mathbf{T}^2]$ (resp. $[\neq^? (\mathbf{T}^1, \ldots, \mathbf{T}^n)]$). A unification constraint is called a *flex-flex pair* if both unification terms have *flexible* heads, i.e., variables at head position.

Clauses consist of disjunctions of literals or unification constraints. The unification constraints specify conditions under which the other literals are valid. For instance the clause $[p_{\alpha\to\beta\to o}\ \mathbf{T}^1_\alpha\ \mathbf{T}^2_\beta]^T \vee [\mathbf{T}^1_\alpha \neq^? \mathbf{S}^1_\alpha] \vee [\mathbf{T}^2_\beta \neq^? \mathbf{S}^2_\beta]$ can be informally read as: *if* $\mathbf{T}^1$ *is unifiable with* $\mathbf{S}^1$ *and* $\mathbf{T}^2$ *with* $\mathbf{S}^2$ *then* $(p\ \mathbf{T}^1\ \mathbf{T}^2)$ *holds*. We implicitly treat the disjunction operator $\vee$ in clauses as commutative and associative, i.e., we abstract from the particular order of the literals. Additionally we presuppose commutativity of $\neq^?$ and implicitly identify any two α-equal constraints or literals. Furthermore we assume that any two clauses have disjoint sets of free variables, i.e., for each freshly generated clause we choose new free variables.

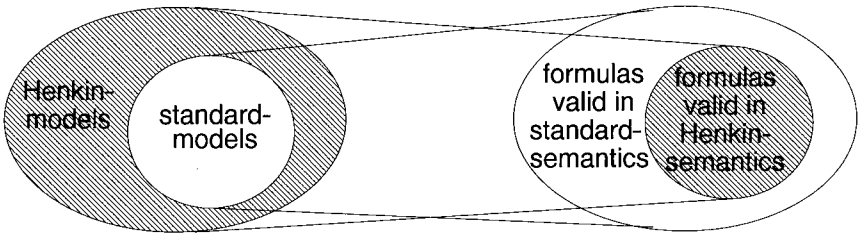
If a clause contains at least one pre-literal we call it a *pre-clause*, otherwise a *proper clause*. A clause is called *empty*, denoted by $\square$, if it consists only of (possibly none) *flex-flex* pairs.

An important aspect of clause normalisation is Skolemisation. In this paper we employ Miller's sound adaptation of traditional first-order Skolemisation (Miller 1983), which associates with each Skolem function the minimum number of arguments the Skolem function has to be applied to. Higher-order Skolemisation becomes sound, if any Skolem function f^n only occurs in a *Skolem term*, i.e., a formula $\mathbf{S} \equiv f^n\overline{\mathbf{A}^n}$, where none of the $\mathbf{A}^i$ contains a bound variable. Thus the Skolem terms only serve as descriptions of the existential witnesses and never appear as functions proper. Without this additional restriction the calculi do not really become unsound, but one can prove an instance of the axiom of choice. Andrews (1973) investigates the following instance: $\exists E_{(\iota\to o)\to o}.\ \forall P_{i\to o}.\ (\exists X_\iota.\ P\ X) \Rightarrow P\ (E\ P))$, which we want to treat as an optional axiom for the resolution calculi presented in this paper; for further details we refer to Miller (1983).

2.3. *Standard and Henkin Semantics*

A *standard model* for $\mathcal{HOL}$ provides a fixed set $\mathcal{D}_\iota$ of individuals, and a set $\mathcal{D}_o := \{\top, \bot\}$ of truth values. The domains for functional types are defined inductively: $\mathcal{D}_{\alpha\to\beta}$ is the set of all functions $f\colon \mathcal{D}_\alpha \to \mathcal{D}_\beta$. *Henkin models*

only require that $\mathcal{D}_{\alpha\to\beta}$ has enough members that any well-formed formula can be evaluated. Thus, the generalisation to Henkin models restricts the set of valid formulas sufficiently, such that complete calculi are possible. The following figure illustrates the sketched connection between standard- and Henkin semantics.



In Henkin and standard semantics Leibniz equality (which is defined as $\doteq^{\alpha} := \lambda X_{\alpha}.\, \lambda Y_{\alpha}.\, \forall P_{\alpha\to o}.\, P\ X \Rightarrow P\ Y$) denotes the intuitive identity relation and the (type parameterised) functional extensionality principles

$$\forall M_{\alpha\to\beta}.\, \forall N_{\alpha\to\beta}.\, (\forall X.\, M\ X \doteq N\ X) \Rightarrow (M \doteq N)$$

as well as the Boolean extensionality principle

$$\forall P_o.\, \forall Q_o.\, (P \Leftrightarrow Q) \Rightarrow (P \doteq Q)$$

are valid (cf. Benzmüller 1999a; Benzmüller and Kohlhase 1997). *Satisfiability* and *validity* ($\mathcal{M} \models \mathbf{F}$ or $\mathcal{M} \models \Phi$) of a formula $\mathbf{F}$ or set of formulas Φ in a model $\mathcal{M}$ are defined as usual.

We want to point out that the above statements on equality and extensionality do not apply to general models as originally introduced by Henkin (1950). Andrews (1972) showed that the sets $\mathcal{D}_{\alpha\to o}$ may be so sparse in Henkin's original notion of general models that Leibniz equality may denote a relation, which does not fulfil the functional extensionality principle. Due to lack of space we cannot present this general model here but refer to Andrews (1972) for further details. The solution suggested by Andrews is to presuppose the presence of the intuitive identity relations in all domains $\mathcal{D}_{\alpha\to\alpha\to o}$, which ensures the existence of unit sets $\{a\} \in \mathcal{D}_{\alpha\to o}$ for all elements $a \in \mathcal{D}_{\alpha}$. The existence of these unit sets in turn ensures that Leibniz equality indeed denotes the intended (fully extensional) identity relation.

In this paper, "Henkin semantics" means the corrected version of Henkin's original notion as given in Andrews (1972).

2.4. *Proving Completeness*

The abstract consistency proof principle (also called unifying principle) is a strong tool supporting the analysis of the connection between syntax and semantics for higher-order calculi. This proof principle has originally been introduced by Smullyan (1963) for first-order logic and has been adapted to higher-order logic by Andrews (1971). However, Andrews' adaptation allows completeness proofs only for the rather weak semantical notion of V-complexes (in which the axioms of extensionality may fail, cf. Benzmüller 1991; Benzmüller and Kohlhase 1997).

The following proof principle adapts Andrews abstract consistency proof principle to Henkin semantics.

DEFINITION 1 (*Acc* for Henkin Models). Let Σ be a signature and Γ_Σ a class of sets of Σ-sentences. If the following conditions hold for all $\mathbf{A}, \mathbf{B} \in cwff_o$, $\mathbf{F}, \mathbf{G} \in cwff_{\alpha\to\beta}$, and $\Phi \in \Gamma_\Sigma$, then we call Γ_Σ an *abstract consistency class for Henkin models*, abbreviated by *Acc*. (We want to point out that we assume an implicit treatment of α-convertibility here, whereas Andrews treats α-convertibility explicit in his notion of η-wffs; cf. Andrews (1971, 3.1.2, 2.7.5).)

saturated $\quad \Phi \cup \{\mathbf{A}\} \in \Gamma_\Sigma$ or $\Phi \cup \{\neg\mathbf{A}\} \in \Gamma_\Sigma$.

∇_c If $\mathbf{A}$ is atomic, then $\mathbf{A} \notin \Phi$ or $\neg\mathbf{A} \notin \Phi$.

$\nabla_\neg$ If $\neg\neg\mathbf{A} \in \Phi$, then $\Phi \cup \{\mathbf{A}\} \in \Gamma_\Sigma$.

∇_β If $\mathbf{A} \in \Phi$ and $\mathbf{B}$ is the β-normal form of $\mathbf{A}$, then $\Phi \cup \{\mathbf{B}\} \in \Gamma_\Sigma$.

∇_η If $\mathbf{A} \in \Phi$ and $\mathbf{B}$ is the η-long form of $\mathbf{A}$, then $\Phi \cup \{\mathbf{B}\} \in \Gamma_\Sigma$.

$\nabla_\vee$ If $\mathbf{A} \vee \mathbf{B} \in \Phi$, then $\Phi \cup \{\mathbf{A}\} \in \Gamma_\Sigma$ or $\Phi \cup \{\mathbf{B}\} \in \Gamma_\Sigma$.

$\nabla_\wedge$ If $\neg(\mathbf{A} \vee \mathbf{B}) \in \Phi$, then $\Phi \cup \{\neg\mathbf{A}, \neg\mathbf{B}\} \in \Gamma_\Sigma$.

$\nabla_\forall$ If $\Pi^\alpha\mathbf{F} \in \Phi$, then $\Phi \cup \{\mathbf{F}\ \mathbf{W}\} \in \Gamma_\Sigma$ for each $\mathbf{W} \in cwff_\alpha$.

$\nabla_\exists$ If $\neg\Pi^\alpha\mathbf{F} \in \Phi$, then $\Phi \cup \{\neg(\mathbf{F}\ w)\} \in \Gamma_\Sigma$ for any new constant $w \in \Sigma_\alpha$.

∇_b If $\neg(\mathbf{A} \doteq^o \mathbf{B}) \in \Phi$, then $\Phi \cup \{\mathbf{A}, \neg\mathbf{B}\} \in \Gamma_\Sigma$ or $\Phi \cup \{\neg\mathbf{A}, \mathbf{B}\} \in \Gamma_\Sigma$.

∇_q If $\neg(\mathbf{F} \doteq^{\alpha\to\beta} \mathbf{G}) \in \Phi$, then $\Phi \cup \{\neg(\mathbf{F}\ w \doteq^\beta \mathbf{G}\ w)\} \in \Gamma_\Sigma$ for any new constant $w \in \Sigma_\alpha$.

This definition extends Andrews notion of *abstract consistency classes for V-complexes* by the new requirements *saturated*, ∇_η, ∇_b, and ∇_q. *Saturatedness* turns the partial V-complexes into total structures and the latter two conditions ensure that Leibniz equality indeed denotes a fully extensional relation (which may not be the case in V-complexes, where Leibniz equality simply not necessarily denotes the intended identity relation; cf. Benzmüller 1991; Benzmüller and Kohlhase 1997).

The following model existence theorem is due to Andrews (1971).

THEOREM 2 (Henkin Model Existence (Andrews 1971)). Let Φ be a set of closed Σ-formulas, Γ_Σ be an abstract consistency class for V-complexes (i.e., Γ_Σ fulfils ∇_c, $\nabla_\neg$, ∇_β, $\nabla_\vee$, $\nabla_\wedge$, $\nabla_\forall$, $\nabla_\exists$), and let $\Phi \in \Gamma_\Sigma$. There exists a V-complex $\mathcal{M}$, such that $\mathcal{M} \models \Phi$.

The following related theorem addressing Henkin semantics (and additional ones addressing several notions in between Henkin semantics and V-complexes) is presented in Benzmüller (1999a); Benzmüller and Kohlhasse (1997).

THEOREM 3 (Henkin Model Existence (Benzmüller and Kohlhase 1998)). Let Φ be a set of closed Σ-formulas, Γ_Σ be an abstract consistency class for Henkin models, and let $\Phi \in \Gamma_\Sigma$. There exists a Henkin model $\mathcal{M}$, such that $\mathcal{M} \models \Phi$.

The complicated task of proving Henkin completeness for a given (resolution) calculus R can now be reduced to showing that the set of all sets Φ containing R-consistent closed formulas is an abstract consistency class for Henkin models, i.e., to verify the (syntactically checkable) conditions given in Definition 1.

3. HIGHER-ORDER RESOLUTION

In this section we introduce several higher-order resolution calculi. Additional approaches not mentioned here are briefly sketched and related to the presented ones in Section 5. The sketched approaches will be compared with respect to their extensionality treatment in Section 4.

3.1. *Andrews' Higher-Order Resolution $\mathcal{R}$*

We transform Andrews' higher-order resolution calculus (Andrews 1971) in our uniform notation. In the remainder of this paper we refer to this calculus with $\mathcal{R}$. Extending Andrews (1971) we show that $\mathcal{R}$ is Henkin

complete if one adds infinitely many extensionality axioms into the search space.

λ-*Conversion.* Calculus $\mathcal{R}$ provides two explicit rules addressing α-conversion and β-reduction (cf. Andrews 1971, 5.1.1) but does not provide a rule for η-conversion. Consequently η-equality of two terms (e.g., $f_{\iota\to\iota} \doteq \lambda X_\iota.\ f\ X$) cannot be proven in this approach without employing the functional extensionality axiom of appropriate type; cf. Section 4.1.

In our presentation we omit explicit rules for α- and β-convertibility and instead treat them implicitly, i.e., we assume that the presented rules operate on input and generate output in β-normal form and we automatically identify terms which differ only with respect to the names of bound variables.

Clause Normalisation. $\mathcal{R}$ introduces only four rules belonging to clause normalisation: negation elimination, conjunction elimination, existential elimination, and universal elimination (cf. Andrews 1971, 5.1.4.–5.1.7.). As our presentation of clauses in contrast to Andrews (1971) explicitly mentions the polarities of clauses and brackets the literal atoms we have to provide additional structural rules, e.g., the rule $\vee^T$.

- Negation elimination: $\dfrac{\mathbf{C} \vee [\neg\mathbf{A}]^T}{\mathbf{C} \vee [\mathbf{A}]^F}\ \neg^T \qquad \dfrac{\mathbf{C} \vee [\neg\mathbf{A}]^F}{\mathbf{C} \vee [\mathbf{A}]^T}\ \neg^F$
- Conjunction[1]/disjunction elimination:

$$\frac{\mathbf{C} \vee [\mathbf{A} \vee \mathbf{B}]^T}{\mathbf{C} \vee [\mathbf{A}]^T \vee [\mathbf{B}]^T}\ \vee^T \qquad \frac{\mathbf{C} \vee [\mathbf{A} \vee \mathbf{B}]^F}{\mathbf{C} \vee [\mathbf{A}]^F}\ \vee_l^F \qquad \frac{\mathbf{C} \vee [\mathbf{A} \vee \mathbf{B}]^F}{\mathbf{C} \vee [\mathbf{B}]^F}\ \vee_r^F$$

- Existential[2]/universal elimination:

$$\frac{\mathbf{C} \vee [\Pi^\alpha \mathbf{A}]^T}{\mathbf{C} \vee [\mathbf{A}\ X_\alpha]^T}\ \Pi^T \qquad \frac{\mathbf{C} \vee [\Pi^\alpha \mathbf{A}]^F}{\mathbf{C} \vee [\mathbf{A}\ \mathrm{s}_\alpha]^F}\ \Pi^F$$

  X_α is a new free variable and s_α is a new Skolem term

Additionally Andrews presents rules addressing commutativity and associativity of the $\vee$-operator connecting the clauses literals (cf. Andrews 1971, 5.1.2.). We have already mentioned the implicit treatment of these aspects in Section 2.2.

In the remainder of this paper Cnf(**A**) denotes the set of clauses obtained from formula **A** by clause normalisation. It is easy to verify that clauses produced with Andrews' original normalisation rules can also be obtained with the rules presented here (and vice versa).

Resolution and Factorisation. Instead of a resolution and a factorisation rule – which work in connection with unification – Andrews presents a simplification and a cut rule. The cut rule is only applicable to clauses with two complementary literals which have identical atoms. Similarly Sim is defined only for clauses with two identical literals. In order to generate identical literal atoms during the refutation process these two rules have to be combined with the substitution rule Sub presented below.

- Simplification:
$$\frac{[\mathbf{A}]^{\mu} \vee [\mathbf{A}]^{\mu} \vee \mathbf{C}}{[\mathbf{A}]^{\mu} \vee \mathbf{C}}\ \text{Sim}$$

- Cut:
$$\frac{[\mathbf{A}]^{\mu} \vee \mathbf{C} \quad [\mathbf{A}]^{\nu} \vee \mathbf{D}}{\mathbf{C} \vee \mathbf{D}}\ \text{Cut}$$

Unification and Primitive Substitution. As higher-order unification was still an open problem in 1971 calculus $\mathcal{R}$ employs the British Museum Method instead, i.e., it provides a substitution rule that allows to blindly instantiate free variables by arbitrary terms. As the instantiated terms may contain logical constants, instantiation of variables in proper clauses may lead to pre-clauses, which must be normalised again with the clause normalisation rules.

- Substitution of arbitrary terms:
$$\frac{\mathcal{C}}{\mathcal{C}_{[\mathbf{T}_\alpha / X_\alpha]}}\ \text{Sub}$$

X_α is a free variable occurring in $\mathcal{C}$.

Extensionality Treatment. Calculus $\mathcal{R}$ does not provide rules addressing the functional and/or Boolean extensionality principles. Instead $\mathcal{R}$ assumes that the following extensionality axioms are (in form of respective clauses) explicitly added to the search space. And since the functional extensionality principle is parameterised over arbitrary functional types infinitely many functional extensionality axioms are required[3].

$$\text{EXT}^{\doteq}_{\alpha\to\beta}\colon \quad \forall F_{\alpha\to\beta}.\, \forall G_{\alpha\to\beta}.\, (\forall X_\beta.\, F\ X \doteq G\ X) \Rightarrow F \doteq G$$
$$\text{EXT}^{\doteq}_{o}\colon \quad \forall A_o.\, \forall B_o.\, (A \Leftrightarrow B) \Rightarrow A \doteq^{o} B$$

These are the crucial directions of the extensionality principles and the backward directions are not needed. The extensionality clauses derived from the extensionality axioms have the following form (note the many free variables, especially at literal head position, that are introduced into the search space – they heavily increase the amount of blind search in any attempt to automate the calculus):

$\mathcal{E}_1^{\alpha\to\beta} : [p\ (F\ s)]^T \vee [Q\ F]^F \vee [Q\ G]^T$ $\qquad$ $\mathcal{E}_1^{o} : [A]^F \vee [B]^F \vee [P\ A]^F \vee [P\ B]^T$
$\mathcal{E}_2^{\alpha\to\beta} : [p\ (G\ s)]^F \vee [Q\ F]^F \vee [Q\ G]^T$ $\qquad$ $\mathcal{E}_2^{o} : [A]^T \vee [B]^T \vee [P\ A]^F \vee [P\ B]^T$

$p_{\beta\to o}$, s_α are Skolem terms and $P_{(\alpha\to\beta)\to o}$, $Q_{(\alpha\to\beta)\to o}$ are new free variables.

Proof Search. Initially the proof problem is negated and normalised. The main proof search then starts with the normalised clauses and applies the cut and simplification rule in close connection with the substitution rule. An intermediate application of the clause normalisation rules may be needed to normalise temporarily generated pre-clauses. The extensionality treatment in $\mathcal{R}$ simply assumes to add at the beginning of the refutation process the above clauses obtained from the extensionality axioms.

When abstracting from the initial and intermediate normalisations the proof search can be illustrated as follows:

Completeness Results. Andrews (1971) gives a completeness proof for calculus $\mathcal{R}$ with respect to the semantical notion of V-complexes. As the extensionality principles are not valid in this rather weak semantical structures, the extensionality axioms are not needed in this completeness proof.

THEOREM 4 (V-completeness of $\mathcal{R}$). The calculus $\mathcal{R}$ is complete with respect to the notion of V-complexes.

Proof. We sketch the proof idea: 4(i) First show that the set of non-refutable sentences in $\mathcal{R}$ is an abstract consistency class for V-complexes. 4(ii) Then prove completeness of $\mathcal{R}$ with respect to V-complexes in an indirect argument: assuming non-completeness of $\mathcal{R}$ leads to an contradiction by 4(i) and Theorem 3. □

We now extend this result and prove Henkin completeness of calculus $\mathcal{R}$.

THEOREM 5 (Henkin completeness of $\mathcal{R}$). The calculus $\mathcal{R}$ is complete with respect to Henkin semantics provided that the infinitely many extensionality axioms are given.

Proof. 5(i) The crucial aspect is to prove that the set of non-refutable sentences in $\mathcal{R}$ enriched by the extensionality axioms is an abstract con-

sistency class for Henkin models. 5(ii) An indirect argument analogous to 4(ii) employing 5(i) and Theorem 3 ensures completeness.

In order to show 5(i) we have to verify the additional abstract consistency properties *saturated*, ∇_η, ∇_b, and ∇_q as specified in Definition 1.

saturated We show that $\Phi \cup \{\mathbf{A}\} \not\vdash_{\mathcal{R}} \square$ or $\Phi \cup \{\neg\mathbf{A}\} \not\vdash_{\mathcal{R}} \square$. Assume $\Phi \not\vdash_{\mathcal{R}} \square$ but $\Phi \cup \{\mathbf{A}\} \vdash_{\mathcal{R}} \square$ and $\Phi \cup \{\neg\mathbf{A}\} \vdash_{\mathcal{R}} \square$. By Lemma 6 (cf. below) we get $\Phi\{\mathbf{A} \vee \neg\mathbf{A}\} \vdash_{\mathcal{ER}} \square$, and hence, since $\mathbf{A} \vee \neg\mathbf{A}$ is a tautology, it must be the case that $\Phi \vdash_{\mathcal{ER}} \square$, which contradicts our assumption.

∇_η Assuming $\mathbf{A} \in \Phi$ and $\Phi \cup \{\mathbf{B}\} \vdash_{\mathcal{R}} \square$, we get $\Phi \vdash_{\mathcal{R}} \square$ by Lemma 7 (cf. below). This ensures the assertion by contraposition.

∇_b We first apply rule *Sub* and instantiate the variables A and B in the Boolean extensionality axioms $\mathcal{E}_1^o$ and $\mathcal{E}_2^o$ with terms $\mathbf{A}$ and $\mathbf{B}$. Now assume that $\neg(\mathbf{A} \doteq^o \mathbf{B}) \in \Phi$ and $\Phi \cup \{\mathbf{A}, \neg\mathbf{B}\} \vdash_{\mathcal{R}} \square$ and $\Phi \cup \{\neg\mathbf{A}, \mathbf{B}\} \vdash_{\mathcal{R}} \square$. Employing the instantiated Boolean extensionality axioms it is easy to see that $\Phi \vdash_{\mathcal{R}} \square$, which ensures the assertion by contraposition.

∇_q Can be shown analogously to ∇_b when appropriately instantiating the functional extensionality axioms $\mathcal{E}_1^{\alpha\to\beta}$, $\mathcal{E}_2^{\alpha\to\beta}$.

LEMMA 6. Let Φ be a set of sentences and $\mathbf{A}, \mathbf{B}$ be sentences. If $\Phi \cup \{\mathbf{A}\} \vdash_{\mathcal{R}} \square$ and $\Phi \cup \{\mathbf{B}\} \vdash_{\mathcal{R}} \square$, then $\Phi \cup \{\mathbf{A} \vee \mathbf{B}\} \vdash_{\mathcal{R}} \square$.

Proof. We first verify that $\mathrm{Cnf}(\Phi * \mathbf{A} \vee \mathbf{B}) = \mathrm{Cnf}(\Phi) \cup (\mathrm{Cnf}(\mathbf{A}) \sqcup \mathrm{Cnf}(\mathbf{B}))$, where $\Gamma \sqcup \Delta = := \{\mathbf{C} \vee \mathbf{D} | \mathbf{C} \in \mathrm{Cnf}(A)\}, \mathbf{D} \in \mathrm{Cnf}(B)\}$. Then we use that $\Phi \cup (\Gamma_1 \sqcup \Gamma_2) \vdash_{\mathcal{R}} \square$, provided that $\Phi \cup \Gamma_1 \vdash_{\mathcal{R}} \square$ and $\Phi \cup \Gamma_2 \vdash_{\mathcal{R}} \square$. $\square$

LEMMA 7. Let Φ be a set of sentences and let $\mathbf{A}, \mathbf{B}$ be sentences in β-normal form, such that $\mathbf{A}$ can be transformed into $\mathbf{B}$ by (i) a one step η-expansion or (ii) a multiple step η-expansion. Then $\Phi \cup \{\mathbf{B}\} \vdash_{\mathcal{R}} \square$ implies $\Phi \cup \{\mathbf{A}\} \vdash_{\mathcal{R}} \square$.

Proof. Case (ii) can be proven by induction on the number of η-expansion steps employing (i) in the base case. To prove case (i) note that $\mathbf{A}$ and $\mathbf{B}$ differ (apart from α-equality) only with respect to a single subterm $\mathbf{T}_{\alpha\to\beta}$. More precisely, $\mathbf{A}_{[(\lambda X_\alpha \mathbf{T}\ X)/\mathbf{T}]}$ is equal to $\mathbf{B}$. Normalising sentences $\mathbf{A}$ (resp. $\mathbf{B}$) may result in several clauses $\mathcal{A}_1, \ldots, \mathcal{A}_n$ (resp. $\mathcal{B}_1, \ldots, \mathcal{B}_n$)

with duplicated occurrences of subterm $\mathbf{T}$ (resp. $\lambda X\mathbf{.}\ \mathbf{T}\ X$). We appropriately instantiate the functional extensionality axioms $\mathcal{E}_1^{\alpha\to\beta}$, $\mathcal{E}_2^{\alpha\to\beta}$ and derive the (Leibniz equation) clauses $\mathcal{C}_1 : [Q\ f]^T \vee [Q\ (\lambda X\mathbf{.}\ f\ X)]^F$ and $\mathcal{C}_2 : [Q'\ f]^F \vee [Q'\ (\lambda X\mathbf{.}\ f\ X)]^T$ (the latter can be obtained from the former by substituting $\lambda X\mathbf{.}\ \neg Q'\ X$ for Q). Obviously, we can derive for each $1 \leq i \leq n$ the clause $\mathcal{B}_i$ from its counterpart $\mathcal{A}_i$ with the help of $\mathcal{C}_1$ and $\mathcal{C}_2$ (formally we apply an induction on the occurrences of term $\mathbf{T}$ in $\mathcal{A}_i$). □

3.2. *Huet's Higher-Order Constrained Resolution $\mathcal{CR}$*

In this section we transform Huet's constrained resolution approach (Huet 1972, 1973a) to our uniform notation. The calculus here is the unsorted fragment of the variant of Huet's approach as presented in Kohlhase (1994). In the remainder of this paper we refer to this calculus as $\mathcal{CR}$. We extend (Huet 1972, 1973a) and show that $\mathcal{CR}$ is Henkin complete if we add infinitely many extensionality axioms to the search space.

λ-Conversion. Like $\mathcal{R}$ calculus $\mathcal{CR}$ assumes that terms, literals, and clauses are implicitly reduced to β-normal form. Furthermore we assume that α-equality is treated implicitly, i.e., we identify all terms that differ only with respect to the names of bound variables.

Clause Normalisation. Huet (1972) does not present clause normalisation rules but assumes that they are given. Here we employ the rules $\neg^T$, $\neg^F$, $\vee^T$, $\vee_l^F$, $\vee_r^F$, Π^T, and Π^F as already defined for calculus $\mathcal{R}$ in Section 3.1.

Resolution and Factorisation. As first-order unification is decidable and unitary it can be employed as a strong filter in first-order resolution (Robinson 1965). Unfortunately higher-order unification is not decidable (cf. Lucchesi 1972; Huet 1973b; Goldfarb 1981) and thus it can not be applied in the sense of a terminating side computation in higher-order theorem proving. Huet therefore suggests in Huet (1972, 1973a) to delay the unification process and to explicitly encode unification problems occurring during the refutation search as unification onstraints. In his original approach Huet presented a hyper-resolution rule which simultaneously resolves on the resolution literals $\mathbf{A}^1, \ldots \mathbf{A}^n$ $(1 \leq n)$ and $\mathbf{B}^1, \ldots \mathbf{B}^m$ $(1 \leq m)$ of two given clauses and adds the unification constraint $[\neq^? (\mathbf{A}^1, \ldots \mathbf{A}^n, \mathbf{B}^1, \ldots \mathbf{B}^m)]$ to the resolvent.

$$\frac{[\mathbf{A}^1]^\mu \vee \ldots \vee [\mathbf{A}^n]^\mu \vee \mathbf{C}[\mathbf{B}^1]^\mu \vee \ldots \vee [\mathbf{B}^m]^\mu \vee \mathbf{D}}{\mathbf{C} \vee \mathbf{D} \vee [\neq^? (\mathbf{A}^1, \ldots \mathbf{A}^n, \mathbf{B}^1, \ldots \mathbf{B}^m)]}\ \text{Hres}$$

In order to ease the comparison with the two other approaches discussed in this paper we instead employ a resolution rule Res and a factorisation rule Fac. Like Hres both rules encode the unification problem to be solved as a unification constraint.

- Constrained resolution:
$$\frac{[\mathbf{A}]^{\mu} \vee \mathbf{C} \quad [\mathbf{B}]^{\nu} \vee \mathbf{D}}{\mathbf{C} \vee \mathbf{D} \vee [\mathbf{A} \neq^{?} \mathbf{B}]} \text{ Res}$$

- Constrained factorisation:
$$\frac{[\mathbf{A}]^{\mu} \vee [\mathbf{B}]^{\mu} \vee \mathbf{C}}{[\mathbf{A}]^{\mu} \vee \mathbf{C} \vee [\mathbf{A} \neq^{?} \mathbf{B}]^{F}} \text{ Fac}$$

One can easily prove by induction on $n + m$ that each proof step applying rule Hres can be replaced by a corresponding derivation employing Res and Fac. For a formal proof note that the unification constraint $[\neq^{?} (\mathbf{A}^1, \ldots \mathbf{A}^n, \mathbf{B}^1, \ldots \mathbf{B}^m)]$ is equivalent to $[\mathbf{A}^1 \neq^{?} \mathbf{A}^2] \vee [\mathbf{A}^2 \neq^{?} \mathbf{A}^3] \vee \ldots \vee [\mathbf{A}^{n-1} \neq^{?} \mathbf{A}^n] \vee [\mathbf{A}^n \neq^{?} \mathbf{B}^1] \vee [\mathbf{B}^1 \neq^{?} \mathbf{B}^2] \vee [\mathbf{B}^2 \neq^{?} \mathbf{B}^3] \vee \ldots \vee [\mathbf{B}^{n-1} \neq^{?} \mathbf{B}^n]$.

Unification and Splitting. Huet (1975) introduces higher-order unification and higher-order pre-unification and shows that higher-order pre-unification is sufficient to verify the soundness of a refutation in which the occurring unification problems have been delayed until the end. The higher-order pre-unification rules presented here are discussed in detail in Benzmüller (1999a). They furthermore closely reflect the rules as presented in Snyder and Gallier (1989).

- Elimination of trivial pairs:
$$\frac{\mathbf{C} \vee [\mathbf{A} \neq^{?} \mathbf{A}]}{\mathbf{C}} \text{ Triv}$$

- Decomposition
$$\frac{\mathbf{C} \vee [\mathbf{A}_{\alpha\to\beta}\, \mathbf{C}_{\alpha} \neq^{?} \mathbf{B}_{\alpha\to\beta}\, \mathbf{D}_{\alpha}]}{\mathbf{C} \vee [\mathbf{A} \neq^{?} \mathbf{B}] \vee [\mathbf{C} \neq^{?} \mathbf{D}]} \text{ Dec}$$

- Elimination of λ-binders:
(weak functional extensionality)
$$\frac{\mathbf{C} \vee [\mathbf{M}_{\alpha\to\beta} \neq^{?} \mathbf{N}_{\alpha\to\beta}]}{\mathbf{C} \vee [\mathbf{M}\, s_{\alpha} \neq^{?} \mathbf{N}\, s_{\alpha}]} \text{ Func}$$
s_{α} is a new Skolem term.

- Imitation of rigid heads:
$$\frac{\mathbf{C} \vee [F_{\gamma}\, \overline{\mathbf{U}^n} \neq^{?} h\, \overline{\mathbf{V}^m}] \quad \mathbf{G} \in \mathcal{GB}^{h}_{\gamma}}{\mathbf{C} \vee [F \neq^{?} \mathbf{G}] \vee [F\, \overline{\mathbf{U}^n} \neq^{?} h\, \overline{\mathbf{V}^m}]} \text{ FlexRigid}$$
$\mathcal{GB}^{h}_{\gamma}$ is the set of partial bindings of type γ for head h as defined in Snyder and Gallier (1989).

Huet points to the usefulness of eager unification to filter out clauses with non-unifiable unification constraints or to back-propagate the solutions of easily solvable constraints (e.g., in case of first-order unification

problems occurring during the proof search). Many of the higher-order unification problems occurring in practice are decidable and have only finitely many solutions. Hence, even though higher-order unification is generally not decidable it is sensible in practice to apply the unification algorithm with a particular resource[4], such that only those unification problems which may have further solutions beyond this bound need to be delayed. In our presentation of calculus $\mathcal{CR}$ we explicitly address the aspect of eager unification and substitution by rule Subst. This rule back-propagates eagerly computed unifiers to the literal part of a clause.

- Eager unification and substitution:

$$\frac{\mathbf{C} \vee [X \stackrel{?}{\neq} \mathbf{A}] \quad X \notin \mathrm{free}(\mathbf{A})}{\mathbf{C}_{[\mathbf{A}/X]}}\ \mathrm{Subst}$$

  Rule Subst is applicable provided that $[X \stackrel{?}{\neq} \mathbf{A}]$ is solved with respect to the other unification constraints in $\mathbf{C}$, i.e., that there is no conflict with other unification constraints.

The literal heads of our clauses may consist of set variables and it may be necessary to instantiate them with terms introducing new logical constant at head position in order to find a refutation. Unfortunately not all appropriate instantiations can be computed with the calculus rules presented so far. To address this problem Huet's approach provides the following splitting rules:

1. Instantiate set variables:

$$\frac{[P\ \mathbf{A}]^T \vee \mathbf{C}}{[Q]^T \vee [R]^T \vee \mathbf{C} \vee [P\ \mathbf{A}\ \stackrel{?}{\neq}\ (Q_o \vee R_o)]}\ \mathrm{S}^T_{\vee}$$

$$\frac{[P\ \mathbf{A}]^\mu \vee \mathbf{C}}{[Q]^\nu \vee \mathbf{C} \vee [P\ \mathbf{A}\ \stackrel{?}{\neq}\ \neg Q_o]}\ \mathrm{S}^{TF}_{\neg} \qquad \frac{[P\ \mathbf{A}]^F \vee \mathbf{C}}{\begin{array}{l}[Q]^F \vee \mathbf{C} \vee [P\ \mathbf{A}\ \stackrel{?}{\neq}\ (Q_o \vee R_o)] \\ [R]^F \vee \mathbf{C} \vee [P\ \mathbf{A}\ \stackrel{?}{\neq}\ (Q_o \vee R_o)]\end{array}}\ \mathrm{S}^F_{\vee}$$

$$\frac{[P\ \mathbf{A}_{\alpha\to o}]^T \vee \mathbf{C}}{[M_{\alpha\to o}\ Z]^T \vee \mathbf{C} \vee [P\ \mathbf{A}\ \stackrel{?}{\neq}\ \Pi^\alpha M]}\ \mathrm{S}^T_{\Pi}$$

$$\frac{[P\ \mathbf{A}_{\alpha\to o}]^F \vee \mathbf{C}}{[M_{\alpha\to o}\ s]^F \vee \mathbf{C} \vee [P\ \mathbf{A}\ \stackrel{?}{\neq}\ \Pi^\alpha M]}\ \mathrm{S}^F_{\Pi}$$

   S^T_{Π} and S^F_{Π} are infinitely branching as they are parameterised over type α. Q_o, R_o, $M_{\alpha\to o}$, Z_α are new variables and s_α is a new Skolem constant.

A theorem which is not refutable in $\mathcal{CR}$ if the splitting rules are not available is $\exists A_o.A$. After negation this statement normalises to clause $\mathcal{C}_1 : [A]^F$, such that none but the splitting rules are applicable. With the help of

rule $S^{TF}_{\neg}$ and eager unification, however, we can derive $\mathcal{C}_2 : [A']^T$ which is then successfully resolvable against $\mathcal{C}_1$.

Extensionality Treatment. On the one hand η-convertibility is built-in in higher-order unification, such that calculus $\mathcal{CR}$ already supports functional extensionality reasoning to a certain extent. On the other hand $\mathcal{CR}$ nevertheless fails to address full extensionality as it does not realise the required subtle interplay between the functional and Boolean extensionality principles. For example, without employing additional Boolean and functional extensionality axioms $\mathcal{CR}$ cannot prove the rather simple Examples presented in Sections 4.2, 4.3, and 4.4.

Proof Search. Initially the proof problem is negated and normalised. The main proof search then operates on the generated clauses by applying the resolution, factorisation, and splitting rules. Despite the possibility of eager unification $\mathcal{CR}$ generally foresees to delay the higher-order unification process in order to overcome the undecidability problem. When deriving an empty clause $\mathcal{CR}$ then tests whether the accumulated unification constraints justifying this particular refutation are solvable. Like $\mathcal{R}$, the extensionality treatment of $\mathcal{CR}$ requires the addition of infinitely many extensionality axioms to the search space. The following figure graphically illustrates the main ideas of the proof search in $\mathcal{CR}$.

Completeness Results. Huet (1972, 1973a) analyses completeness of $\mathcal{CR}$ only with respect to Andrews V-complexes, i.e., Huet verifies that the set of non-refutable sentences in $\mathcal{CR}$ is an abstract consistency class for V-complexes.

THEOREM 8 (V-completeness of $\mathcal{CR}$). The calculus $\mathcal{CR}$ is complete with respect to the notion of V-complexes.

We now extend this result and prove Henkin completeness of calculus $\mathcal{CR}$.

THEOREM 9 (Henkin completeness of $\mathcal{CR}$). The calculus $\mathcal{CR}$ is complete wrt. Henkin semantics provided that the infinitely many extensionality axioms are given.

Proof. Analogously to the proof of Theorem 5 we can reduce the problem to verifying that the set of non-refutable sentences in $\mathcal{R}$ enriched by the extensionality axioms is an abstract consistency class for Henkin models. The assertion then follows in an indirect argument employing Theorem 3. In addition to the abstract consistency properties already examined in Huet (1972, 1973a) for Theorem 8 we have to verify *saturatedness*, ∇_η, ∇_b, and ∇_q as specified in Definition 1. The proofs of all four statements are analogous to the corresponding parts in the proof of Theorem 5. For *saturatedness* and ∇_η we use analogues of Lemmas 6 and 7.

LEMMA 10. Let Φ be a set of sentences and $\mathbf{A}, \mathbf{B}$ be sentences. If $\Phi \cup \{\mathbf{A}\} \vdash_{\mathcal{CR}} \square$ and $\Phi \cup \{\mathbf{B}\} \vdash_{\mathcal{CR}} \square$, then $\Phi \cup \{\mathbf{A} \vee \mathbf{B}\} \vdash_{\mathcal{CR}} \square$

Proof. Analogous to the proof of Lemma 6. □

LEMMA 11. Let Φ be a set of sentences and let $\mathbf{A}, \mathbf{B}$ be sentences in β-normal form, such that $\mathbf{A}$ can be transformed into $\mathbf{B}$ by (i) a one step η-expansion or (ii) a multiple step η-expansion. Then $\Phi \cup \{\mathbf{B}\} \vdash_{\mathcal{CR}} \square$ implies $\Phi \cup \{\mathbf{A}\} \vdash_{\mathcal{CR}} \square$.

Proof. The proof is analogous to Lemma 7. The main difference is with regard to the derivability of the clauses $\mathcal{B}_i$ from its counterparts $\mathcal{A}_i$ with the help of $\mathcal{C}_1$ and $\mathcal{C}_2$ obtained from the (suitably instantiated) functional extensionality axioms. It might be the case that the terms $\mathbf{T}$ occur inside flexible literals of the clauses $\mathcal{A}_i$. Resolving these flexible literals against $\mathcal{C}_1$ and $\mathcal{C}_2$ results then in flex-flex pairs that cannot be solved eagerly but have to be delayed. E.g., let $\mathcal{A}_j$ $(1 \leq j \leq n)$ be of form $[R\ (p\ \mathbf{T})]^\nu \vee \mathcal{D}$. Instead of $\mathcal{B}_j := [R\ (p\ (\lambda X_{\bullet}\ \mathbf{T}\ X))]^\nu \vee \mathcal{D}$ we can derive only $\mathcal{B}'_j := [Q\ (\lambda X_{\bullet}\ \mathbf{T}\ X)]^\nu \vee \mathcal{D} \vee [Q\ T \neq^?\ R\ (p\ (\lambda X_{\bullet}\ \mathbf{T}\ X))]$. Hence, we have to show (in a technically rather complicated inductive proof on the length of the derivation) that each refutation employing $\mathcal{B}'_j$ can be replaced by a corresponding one employing $\mathcal{B}_j$. □

3.3. *Higher-Order E-Resolution CRE*

Some more recent approaches to higher-order theorem proving employ equational higher-order unification instead of syntactical higher-order unification in order to ease and shorten proofs on the resolution layer by relocating particular computation or reasoning tasks to the unification process. For instance, equational higher-order unification has been investigated within the contexts of higher-order rewriting and narrowing (cf. Nipkow and Prehofer 1998; Prehofer 1998), and within the context of restricted higher-order E-resolution (Wolfram 1993).

In this Section we will sketch a higher-order E-resolution approach based on calculus $\mathcal{CR}$. In contrast to the other investigated calculi the aim thereby is not to provide a detailed description of the particular rules and the functioning of the calculus, but to provide a sufficient basis for the investigation to what extent equational higher-order unification can improve the extensionality reasoning in a higher-order theorem prover.

Generally unification of two (or several) terms $\mathbf{S}$ and $\mathbf{T}$ aims at computing sets of unifiers, i.e., substitutions σ, such that $\mathbf{S}_\sigma$ equals $\mathbf{T}_\sigma$ ($\mathbf{S}_\sigma = \mathbf{T}_\sigma$). Equational unification thereby extends syntactical unification in the sense that it tries to equalise $\mathbf{S}_\sigma$ and $\mathbf{T}_\sigma$ modulo a fixed equational theory E (written as $\mathbf{S}_\sigma =_E \mathbf{T}_\sigma$) instead of equalising them syntactically. A survey to unification theory is given in Baader and Siekmann (1994), and Siekmann (1989).

Within our higher-order context we assume that an equational theory E is defined by a fixed set of equations between closed λ-terms. For instance, equations expressing commutativity and associativity of the $\wedge$-operator are $(\lambda X_o.\, \lambda Y_o.\, X \wedge Y) = (\lambda X_o.\, \lambda Y_o.\, Y \wedge X)$ and $(\lambda X_o.\, \lambda Y_o.\, \lambda Z_o.\, (X \wedge Y) \wedge Z) = (\lambda X_o.\, \lambda Y_o.\, \lambda Z_o.\, X \wedge (Y \wedge Z))$.

And within this particular theory E (to be more precise modulo the congruence relation defined by this equations) the following two terms are unifiable by $[a/X]$: $(p_{o\to o}\ (b_o \wedge X_o) \wedge (X_o \wedge b_o))$ and $(p_{o\to o}\ a_o \wedge (a_o \wedge (b_o \wedge b_o)))$.

We want to point out that Huet's unification approach as presented for calculus $\mathcal{CR}$ is of course not a pure syntactical one as it already takes $\alpha\beta\eta$-equality into account. We nevertheless call Huet's approach *syntactical higher-order unification* in this paper in order to distinguish it from equational higher-order unification in the sense of this Subsection, where the theory E may contain additional higher-order equations.

Several, often restricted, approaches to higher-order E-unification have been discussed in literature. Wolfram (1993) a general higher-order E-unification approach which employs higher-order rewriting techniques. An approach restricted to first-order theories is given in Snyder (1990) and another restricted one, where as much computation as possible is pushed to a first-order E-unification procedure, is discussed in Qian and Wang (1996) and Nipkow and Qian (1991). Dougherty and Johann (1992) presents a restricted combinatory logic approach.

We now sketch our higher-order E-resolution approach $\mathcal{CRE}$.

Clause Normalisation, Resolution and Factorisation, and Splitting. We assume that calculus $\mathcal{CRE}$ coincides with calculus $\mathcal{CR}$ in all but the uni-

fication part. Thus $\mathcal{CR}$ provides the clause normalisation, resolution and factorisation, and splitting rules as introduced in Section 3.2.

Equational Unification. Instead of presenting a concrete set of rules for higher-order E-unification we refer to the respective approaches given in Snyder (1990), Nipkow and Qian (1991), Wolfram (1993), and Qian and Wang (1996). For our investigation of $\mathcal{CRE}$ it will be of minor importance which particular approach we choose and how general this approach is.

Whereas higher-order E-unification can indeed partially improve the extensionality treatment in $\mathcal{CRE}$, we will present simple theorems in Section 4 which cannot be proven in $\mathcal{CRE}$ (or in any of the related approaches mentioned above) without additional extensionality axioms. These counterexamples do not depend on the concrete choice of an equational theory E.

3.4. *Extensional Higher-Order Resolution $\mathcal{ER}$*

We now present the extensional higher-order resolution approach as introduced in Benzmüller and Kohlhase (1998a), Benzmüller (1991a). In the remainder of this paper we refer to this calculus as $\mathcal{ER}$. $\mathcal{ER}$ is Henkin complete without requiring additional extensionality axioms.

λ-Conversion. In contrast to $\mathcal{R}$ and $\mathcal{CR}$ calculus $\mathcal{ER}$ assumes that all terms, literals, and clauses are implicitly reduced to long $\beta\eta$-normal form.

Clause Normalisation, Resolution and Factorisation, and Unification and Splitting. $\mathcal{ER}$ employs the normalisation rules $\neg^T$, $\neg^F$, $\vee^T$, $\vee^F_l$, $\vee^F_r$, Π^T, Π^F, the resolution and factorisation rules Res, Fac, and the unification rules Triv, Dec, Func, FlexRigid, Subst as already defined for calculus $\mathcal{CR}$ in Section 3.2.

Additionally $\mathcal{ER}$ employs the infinitely branching unification rule FlexFlex, which guesses instances in case of flex-flex pairs (cf. Conjecture 13 in Section 3.4).

- Guess
$$\frac{\mathbf{C} \vee [F_{\overline{\gamma^n}\to\alpha}\ \overline{\mathbf{U}^n} = H_{\overline{\delta^m}\to\alpha}\ \overline{\mathbf{V}^m}]^F \quad \mathbf{G} \in \mathcal{GB}^h_{\overline{\gamma^n}\to\alpha}}{\mathbf{C} \vee [F\ \overline{\mathbf{U}^n} = H\ \overline{\mathbf{V}^m}]^F \vee [F = \mathbf{G}]^F}\ \mathit{FlexFlex}$$
$\mathcal{GB}^h_{\overline{\gamma^n}\to\alpha}$ is the set of partial bindings of type γ for a constant h in the given signature.

The splitting rules presented for $\mathcal{CR}$ in Section 3.2 are replaced in $\mathcal{ER}$ by the more elegant primitive substitution rule as first introduced by Andrews (1989).

- Primitive substitution
$$\frac{[Q_\gamma\ \overline{\mathbf{U}^k}]^\alpha \vee \mathbf{C} \quad \mathbf{P} \in \mathcal{GB}_\gamma^{\{\neg,\vee\}\cup\{\Pi^\beta|\beta\in\mathcal{T}\}}}{[Q_\gamma\ \overline{\mathbf{U}^k}]^\alpha \vee \mathbf{C} \vee [Q = \mathbf{P}]^F}\ \mathit{Prim}$$

  $\mathcal{GB}_\gamma^{\{\neg,\vee\}\cup\{\Pi^\beta|\beta\in\mathcal{T}\}}$ is the set of partial bindings of type o for a logical constant in the signature.

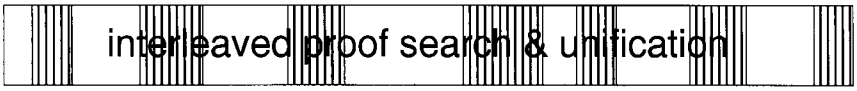
Extensionality Treatment. Instead of adding infinitely many extensionality axioms to the search space $\mathcal{CR}$ provides two new extensionality rules which closely connect refutation search and eager unification. The idea is to allow for recursive calls from higher-order unification to the overall refutation process. This turns the rather weak syntactical higher-order unification approach considered so far into a most general approach for *dynamic* higher-order theory unification.

- Unification and equivalence:
$$\frac{\mathbf{C} \vee [\mathbf{M}_o \neq^? \mathbf{N}_o]}{\mathbf{C} \vee [\mathbf{M}_o \Leftrightarrow \mathbf{N}_o]^F}\ \text{Equiv}$$
- Unification and Leibniz equality:
$$\frac{\mathbf{C} \vee [\mathbf{M}_\alpha \neq^? \mathbf{N}_\alpha]}{\mathbf{C} \vee [\forall P_{\alpha\to o}.\ P\ \mathbf{M} \Rightarrow P\ \mathbf{N}]^F}\ \text{Leib}$$

Proof Search. Initially the proof problem is negated and normalised. The main proof search then closely interleaves the refutation process on resolution layer and unification, i.e., the main proof search rules Res, Fac, and Prim and the unification rules are integrated at a common conceptual level. The calls from unification to the overall refutation process with rules Leib and Equiv introduce new clauses into the search space which can be resolved against already given ones.

This close interplay between unification and refutation search compensates the infinitely many extensionality axioms required in $\mathcal{R}$ and $\mathcal{CR}$ by a more goal-directed approach to full extensionality reasoning.

The following picture graphically illustrates the main ideas of the proof search in $\mathcal{ER}$.



Completeness Results. Henkin completeness of the presented approach with rule FlexFlex is analysed in detail in Benzmüller (1999a) and Benzmüller and Kohlhase (1998a). Here we only mention the main result:

THEOREM 12 (Henkin completeness of $\mathcal{ER}$). The calculus $\mathcal{ER}$ is complete with respect to Henkin semantics.

Benzmüller (1999a) presents but does not prove the following interesting claims which are of major practical importance as they will lead to an enormous reduction of the search spaces in $\mathcal{ER}$.

CONJECTURE 13 (FlexFlex-rule is not needed). Rule FlexFlex can be avoided in $\mathcal{ER}$ without affecting Henkin completeness.

CONJECTURE 14 (Base type restriction of rule Leib). Rule Leib can be restricted to base types α in $\mathcal{ER}$ without affecting Henkin completeness.

4. EXAMPLES

In this section we compare the extensionality treatment provided by the calculi $\mathcal{R}$, $\mathcal{CR}$, $\mathcal{CRE}$, and $\mathcal{ER}$ with the help of simple examples. Despite their simplicity the latter two of these examples are nevertheless challenging with respect to their automisation in a higher-order theorem prover.

4.1. *η-Equality*

EXAMPLE 15. $f_{\iota\to\iota} \doteq \lambda X_\iota.\ f\ X$

Solution in $\mathcal{R}$. In order to prove Example 15, which normalises after negation and expansion of Leibniz equality to $\mathcal{C}_1 : [q\ f]^F$ and $\mathcal{C}_2 : [q\ (\lambda X.\ f\ X)]^T$ where $q_{(\iota\to\iota)\to o}$ is a new Skolem term, we first have to appropriately instantiate the two functional extensionality clauses $\mathcal{E}_1^{\alpha\to\beta}$ and $\mathcal{E}_2^{\alpha\to\beta}$ with the help of rule Sub:

$$\mathcal{E}_1^{\iota\to\iota} : [p\ (f\ s)]^T \vee [Q\ f]^F \vee [Q\ (\lambda X.\ f\ X)]^T$$
$$\mathcal{E}_2^{\iota\to\iota} : [p\ (f\ s)]^F \vee [Q\ f]^F \vee [Q\ (\lambda X.\ f\ X)]^T$$

Employing cut and simplification we can derive

$$\mathcal{C}_3 : [Q\ f]^F \vee [Q\ (\lambda X.\ f\ X)]^T$$

which corresponds to the Leibniz equation between f and $(\lambda X.\ f\ X)$. With rule *Sub* we then substitute the term $\lambda M_{\iota\to\iota}.\ \neg(q\ M)$ for the predicate variable Q, re-normalise the generated pre-clause, and obtain

$$\mathcal{C}_4 : [q\ f]^T \vee [q\ (\lambda X.\ f\ X)]^F$$

By applying the cut rule to $\mathcal{C}_4$, $\mathcal{C}_1$, and $\mathcal{C}_2$ we then derive $\square$.

Solution in $\mathcal{CR}$, $\mathcal{CRE}$, and $\mathcal{ER}$. We first sketch the proof of Example 15 in $\mathcal{CR}$. Initially we resolve on $\mathcal{C}_1 : [q\ f]^F$ and $\mathcal{C}_2 : [q\ (\lambda X.\ f\ X)]^T$ and thereby obtain the unification constraint $\mathcal{C}_3 : \square \vee [f \neq^? (\lambda X.\ f\ X)]^F$. The η-equality of the two unification terms is shown with the help of the unification rule *Func* which derives the trivial unification constraint $\mathcal{C}_4 : \square \vee [f\ s \neq^?\ f\ s]^F$ (where s_ι is new Skolem term). This unification constraint can be subsequently eliminated with rule Triv. Our examples illustrates higher-order unification already addresses weak functional extensionality (η-equality).

An analogous refutation can clearly be employed in calculus $\mathcal{CRE}$ as weak functional extensionality is built-in in higher-order E-unification as well.

Example 15 is trivially solvable in $\mathcal{ER}$ due to the fact that we implicitly assume all terms to be in long $\beta\eta$-normal form, i.e., the clauses to be refuted are $\mathcal{C}_1 : [q(\lambda X.\ fX)]^F$ and $\mathcal{C}_2 : [q(\lambda X.\ fX)]^T$. Clearly, when considering long $\beta\eta$-normal forms instead of β-normal forms the problem is trivially solvable in calculi $\mathcal{R}$, $\mathcal{CR}$, and $\mathcal{CRE}$ as well.

4.2. *Set Descriptions*

In higher-order logic sets can be elegantly encoded by characteristic functions. An interesting problem then is to investigate whether two encodings describe the same set. The following trivial example demonstrates the importance of the extensionality principles for this purpose.

EXAMPLE 16. The set of all red balls equals the set of all balls that are red: $\{X|\text{red}\ X \wedge \text{ball}\ X\} = \{X|\text{ball}\ X \wedge \text{red}\ X\}$. This problem can be encoded as $(\lambda X_\iota.\ \text{red}\ X \wedge \text{ball}\ X) = (\lambda X_\iota.\ \text{ball}\ X \wedge \text{red}\ X)$.

Negation, expansion of Leibniz equality, and clause normalisation leads to the following clauses (where $p_{(\iota\to o)\to o}$ is a new Skolem constant):

$$\mathcal{C}_1 : [p\ (\lambda X.\ \text{red}\ X \wedge \text{ball}\ X)]^F \qquad \mathcal{C}_2 : [p\ (\lambda X.\ \text{ball}\ X \wedge \text{red}\ X)]^T$$

Solution in $\mathcal{R}$. As no rule is applicable to $\mathcal{C}_1$ and $\mathcal{C}_2$ Example 16 is not refutable in $\mathcal{R}$ without employing extensionality axioms. The only way to derive a contradiction is to employ suitable instances of the extensionality clauses in a rather complicated derivation:

1. With rule Sub instantiate the Boolean extensionality axioms $\mathcal{E}_1^o$ and $\mathcal{E}_2^o$ with the terms (red $Y \wedge$ ball Y) and (ball $Y \wedge$ red Y) for variables A

and B. By normalising and employing simplification exhaustively to the resulting pre-clauses we obtain among others:

$$\begin{array}{lllll} \mathcal{C}_3: & [\text{red } Y]^F \vee & [\text{ball } Y]^F \vee & [P\ \mathbf{F}^1]^F \vee & [P\ \mathbf{G}^1]^T \\ \mathcal{C}_4: & [\text{red } Y]^T \vee & & [P\ \mathbf{F}^1]^F \vee & [P\ \mathbf{G}^1]^T \\ \mathcal{C}_5: & & [\text{ball } Y]^T \vee & [P\ \mathbf{F}^1]^F \vee & [P\ \mathbf{G}^1]^T \end{array}$$

where $\mathbf{F}^1$ stands for the term (red $Y \wedge$ ball Y) and $\mathbf{G}^1$ for (ball $Y \wedge$ red Y).

From $\mathcal{C}_3$–$\mathcal{C}_5$ we derive $\mathcal{C}_6 : [P\ \mathbf{F}^1]^F \vee [P\ \mathbf{G}^1]^T$ by cut and simplification, where $\mathcal{C}_6$ corresponds to the clause normal form of $\forall Y.\ ((\lambda X.\ \text{red } X \wedge \text{ball } X)\ Y) \doteq ((\lambda X.\ \text{ball } X \wedge \text{red } X)\ Y)$.

2. With rule Sub we now instantiate the functional extensionality axioms $\mathcal{E}_1^{\iota\to o}$ and $\mathcal{E}_2^{\iota\to o}$ with terms $\mathbf{F}^2 := (\lambda X.\ \text{red } X \wedge \text{ball } X)$ for variable F and $\mathbf{G}^2 := (\lambda X.\ \text{ball } X \wedge \text{red } X)$ for variable G.

$$\mathcal{C}_7 : [q\ (\text{red } s \wedge \text{ball } s)]^T \vee [Q\ \mathbf{F}^2]^F \vee [Q\ \mathbf{G}^2]^T$$
$$\mathcal{C}_8 : [q\ (\text{ball } s \wedge \text{red } s)]^F \vee [Q\ \mathbf{F}^2]^F \vee [Q\ \mathbf{G}^2]^T$$

3. Applying substitution $[(\lambda Z.\ q\ Z)/P, s/Y]$ with rule Sub to clause $\mathcal{C}_6$ leads to:

$$\mathcal{C}_9 : [q\ (\text{red } s \wedge \text{ball } s)]^F \vee [q\ (\text{ball } s \wedge \text{red } s)]^T$$

Applying cut and simplification we combine the results of the above steps and derive from $\mathcal{C}_7$, $\mathcal{C}_8$, and $\mathcal{C}_9$

$$\mathcal{C}_{10} : [Q\ (\lambda X.\ \text{red } X \wedge \text{ball } X)]^F \vee [Q\ (\lambda X.\ \text{ball } X \wedge \text{red } X)]^T$$

which represent the Leibniz equation between $(\lambda X.\ \text{red } X \wedge \text{ball } X)$ and $(\lambda X.\ \text{ball } X \wedge \text{red } X)$. With the help of $\mathcal{C}_1$ and $\mathcal{C}_2$ we can now derive $\square$ after appropriately instantiating $\mathcal{C}_{10}$ with $[p/Q]$.

Note that in Steps 1 and 2 we had to guess the *right* instantiations of the extensionality axioms and to apply non-goal directed forward reasoning.

Solution in $\mathcal{CR}$. The only rule that is applicable to $\mathcal{C}_1$ and $\mathcal{C}_2$ in calculus $\mathcal{CR}$ is the resolution rule Res leading to the following unification constraint

$$\mathcal{C}_3 : \square \vee [p\ (\lambda X.\ \text{red } X \wedge \text{ball } X) \neq^?\ p\ (\lambda X.\ \text{ball } X \wedge \text{red } X)]$$

As this unification constraint is obviously not solvable by syntactical higher-order unification we cannot find a refutation on this derivation path.

As in calculus $\mathcal{R}$ the only way to find a refutation is to guess appropriate instances of the extensionality axioms and to derive from them clause $\mathcal{C}_{10}$ representing the Leibniz equation between $(\lambda X.\ \text{red } X \wedge \text{ball } X)$ and

$(\lambda X_{\cdot}\ \text{ball}\ X \wedge \text{red}\ X)$. A concrete derivation can be carried out analogously to the above derivation in $\mathcal{R}$. The only difference is that we employ resolution and factorisation instead of cut and simplification. In contrast to $\mathcal{R}$ we thereby gain additional guidance with respect to finding some of the required instantiations when combining the resolution/factorisation steps with eager unification attempts. But note that this only holds for the instantiation of non-formulas, e.g., as given in Step 3. The key step in the proof, namely the instantiation of the extensionality axioms in Step 1 with appropriate formulas as arguments, is not supported by unification. Instead the splitting rules have to be employed in order to guess the right instances. The problem with the splitting rules (or analogously the primitive substitution rule) is that each application introduces new clauses with flexible literals into the search space (in case of S^T_Π and S^F_Π even infinitely many) such that the splitting rules become recursively applicable to the new clauses as well.

Consequently, the extensionality treatment in $\mathcal{CR}$ is analogously to the one in $\mathcal{R}$ rather hard to guide in practice. Overwhelming the search space with extensionality clauses and applying forward reasoning to them furthermore principally contrasts the intended character of resolution based theorem proving.

Solution in $\mathcal{CRE}$. Analogous to the unsuccessful initial attempt in $\mathcal{CR}$ we first resolve between $\mathcal{C}_1$ and $\mathcal{C}_2$ and obtain

$$\mathcal{C}_3 : \Box \vee [p\ (\lambda X_{\cdot}\ \text{red}\ X \wedge \text{ball}\ X) \neq^{?} p\ (\lambda X_{\cdot}\ \text{ball}\ X \wedge \text{red}\ X)]$$

Whereas syntactical unification as employed in $\mathcal{CR}$ clashes on this unification constraint, calculus $\mathcal{CRE}$ can solve this E-unification problem provided that the employed E-unification algorithm covers associativity of the $\wedge$-operator (i.e., $E \models (\lambda X_o{\cdot}\ \lambda Y_o{\cdot}\ X \wedge Y) = (\lambda X_o{\cdot}\ \lambda Y_o{\cdot}\ Y \wedge X)$).

Hence, depending on the peculiarity of unification theory E calculus $\mathcal{CRE}$ can provide more goal directed solutions to particular examples and avoid applications of the extensionality axioms. However, the examples below will demonstrate that E-unification does not provide a general solution.

Solution in $\mathcal{ER}$. Calculus $\mathcal{ER}$ provides another goal directed solution avoiding the extensionality axioms. Instead of employing equational unification calculus $\mathcal{ER}$ analyses the unifiability of the unification constraint $\mathcal{C}_3$ with the help of a recursive call from within its unification algorithm to its own overall refutation process. Clearly, this idea can be seen as a very general form of equational unification, namely equational unification

modulo the theory defined by the given clause context and full higher-order logic.

Like above we initially resolve between $\mathcal{C}_1$ and $\mathcal{C}_2$ and obtain clause $\mathcal{C}_3$. Then we transform $\mathcal{C}_3$ with the unification rules Dec and Func into

$$\mathcal{C}_4 : \Box \vee [\text{red } s \wedge \text{ball } s \neq^{?} \text{ball } s \wedge \text{red } s]$$

and apply a recursive call to the overall refutation process with the Boolean extensionality rule Equiv. After normalisation and elimination of identical literals we thereby obtain the following trivially refutable set of propositional clauses

$$\mathcal{C}_5 : [\text{red } s]^F \vee [\text{ball } s]^F \qquad \mathcal{C}_6 : [\text{red } s]^T \qquad \mathcal{C}_7 : [\text{ball } s]^T$$

4.3. *Reasoning with Classical Logic*

The following theorem states that all unary logical operators $O_{o\to o}$ which map the propositions a and b to $\top$ consequently also map $a \wedge b$ to $\top$.

EXAMPLE 17. $\forall O_{o\to o}.\ (O\ a_o) \wedge (O\ b_o) \Rightarrow (O\ (a_o \wedge b_o))$.

Negation and normalisation leads to ($o_{o\to o}$ is a Skolem constant for O)

$$\mathcal{C}_1 : [o\ a]^T \qquad \mathcal{C}_2 : [o\ b]^T \qquad \mathcal{C}_3 : [o\ (a \wedge b)]^F$$

Solution in $\mathcal{R}$. Obviously there is no rule applicable to $\mathcal{C}_1$ – $\mathcal{C}_3$. As in Section 4.2 we are forced to appropriately instantiate the extensionality axioms. In particular we employ the following two instantiations of the Boolean extensionality principle $\text{EXT}^{\doteq}_o$:

$$(a \Leftrightarrow (a \wedge b)) \Leftrightarrow (a \doteq^o (a \wedge b))$$

and

$$(b \Leftrightarrow (a \wedge b)) \Leftrightarrow (b \doteq^o (a \wedge b))$$

That means we guess the substitutions $[a/A, (a \wedge b)/B]$, $[b/A, (a \wedge b)/B]$ and then instantiate the Boolean extensionality clauses $\mathcal{E}^o_1$ and $\mathcal{E}^o_2$ with rule Sub. From the instantiated clauses we can now derive

$$\mathcal{C}_4 : [P\ a]^F \vee [P\ (a \wedge b)]^T \vee [Q\ b]^F \vee [Q\ (a \wedge b)]^T$$

which represents that $(a \doteq (a \wedge b)) \vee (b \doteq (a \wedge b))$. By instantiating P and Q with o and simplification we obtain:

$$\mathcal{C}_5 : [o\ a]^F \vee [o\ b]^F \vee [o\ (a \wedge b)]^T$$

Resolving against $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$ leads to $\square$.

Solution in $\mathcal{CR}$ and $\mathcal{CRE}$. There are only two possible proof steps at the very beginning: resolve between $\mathcal{C}_1$ and $\mathcal{C}_3$ and between $\mathcal{C}_2$ and $\mathcal{C}_3$. Thereby we get

$$\mathcal{C}_4 : \square \vee [p\ a \neq^?\ p\ (a \wedge b)] \quad \mathcal{C}_5 : \square \vee [p\ b \neq^?\ p\ (a \wedge b)]$$

Both unification constraints are neither solvable by syntactical higher-order unification nor by higher-order E-unification.

Successful refutations in $\mathcal{CR}$ and $\mathcal{CRE}$ therefore require the application of appropriately instantiated extensionality clauses as demonstrated within the refutation in calculus $\mathcal{R}$ above. Note that higher-order (E-)unification does not even provide any support for choosing the *right* instantiations of the extensionality axioms.

Hence both calculi, $\mathcal{CR}$ as well as $\mathcal{CRE}$, cannot be Henkin complete without additional extensionality axioms.

Solution in $\mathcal{ER}$. $\mathcal{ER}$ allows for a straightforward refutation of the clauses $\mathcal{C}_1 - \mathcal{C}_3$. Like in $\mathcal{CR}$ and $\mathcal{CRE}$ the only possible steps at the beginning are to resolve between $\mathcal{C}_1$ and $\mathcal{C}_3$ and between $\mathcal{C}_2$ and $\mathcal{C}_3$. Thereby we get

$$\mathcal{C}_4 : \square \vee [p\ a \neq^?\ p\ (a \wedge b)] \quad \mathcal{C}_5 : \square \vee [p\ b \neq^?\ p\ (a \wedge b)]$$

Decomposing both the unification constraints in both clauses leads to

$$\mathcal{C}_6 : \square \vee [a \neq^?\ (a \wedge b)] \quad \mathcal{C}_7 : \square \vee [b \neq^?\ (a \wedge b)]$$

When regarding both unification constraints isolated they are obviously neither syntactically nor semantically solvable. When considering them simultaneously, however, it is easy to see that at least one of both unification constraints must be solvable. Such a *non-constructive* reasoning on the simultaneous solvability/non-solvability of unification constraints is handled in $\mathcal{ER}$ by recursive calls from unification to the overall proof search. In this sense $\mathcal{ER}$ intuitively first assumes that the unification constraints are simultaneously not solvable and then tries to refute this assumption. More concretely, the recursive calls with rule Equiv applied

to $\mathcal{C}_6$ and $\mathcal{C}_7$ introduce after normalisation and factorisation the following clauses into the search space (note the importance of the fact that the generated clauses are analysed in a common context):

$$\mathcal{C}_5 : [a]^F \vee [b]^F \quad \mathcal{C}_6 : [a]^T \vee [b]^T \quad \mathcal{C}_7 : [a]^T \quad \mathcal{C}_8 : [b]^T$$

Clauses $\mathcal{C}_5$–$\mathcal{C}_8$ can be refuted immediately, which contradicts the assumption of the simultaneous semantical non-unifiability of the unification constraints in $\mathcal{C}_6$ and $\mathcal{C}_7$. Hence, either $\mathcal{C}_6$ or $\mathcal{C}_7$ must already be the empty clause, which justifies the proof.

4.4. *Mappings from Booleans to Booleans*

We already mentioned in Section 2.3 that in Henkin semantics the domain $\mathcal{D}_o$ of all Booleans contains exactly the truth values $\bot$ and $\top$. Consequently the domain of all mappings from Booleans to Booleans contains exactly[5] the denotations of the following four functions: $\lambda X_o.\, X_o$, $\lambda X_o.\, \neg X_o$, $\lambda X_o.\, \bot$, and $\lambda X_o.\, \top$. This theorem can be formulated as follows (where $f_{o\to o}$ is a constant):

$$(f = \lambda X_o.\, X_o) \vee (f = \lambda X_o.\, \neg X_o) \vee (f = \lambda X_o.\, \bot) \vee (f = \lambda X_o.\, \top)$$

By unfolding the definition of Leibniz equality, negating the theorem, and applying clause normalisation we obtain the following clauses (where $p^1, \ldots, p^4$ are Skolem constants):

$$\mathcal{D}_1 : [p^1\ f]^T \quad \mathcal{D}_2 : [p^1\ \lambda X_o.\, X_o]^F \quad \mathcal{D}_3 : [p^2\ f]^T \quad \mathcal{D}_4 : [p^2\ \lambda X_o.\, \neg X_o]^F$$

$$\mathcal{D}_5 : [p^3\ f]^T \quad \mathcal{D}_6 : [p^3\ \lambda X_o.\, \bot]^F \quad \mathcal{D}_7 : [p^4\ f]^T \quad \mathcal{D}_8 : [p^4\ \lambda X_o.\, \top]^F$$

Solution in $\mathcal{R}$, $\mathcal{CR}$, and $\mathcal{CRE}$. As the reader may easily check, none of the applicable resolution steps leads to a unification constraint that is solvable by higher-order unification or higher-order E-unification (independent from theory E).

In order to find a refutation appropriate instances of the extensionality principles are needed, just as illustrated in the previous example. Because of lack of space we do not present the quite lengthy refutation here.

Solution in $\mathcal{ER}$. In $\mathcal{ER}$ we can find the following goal directed refutation of the clauses $\mathcal{D}_1, \ldots, \mathcal{D}_8$. We first resolve between the related clauses $\mathcal{D}_1$ and $\mathcal{D}_2$, $\mathcal{D}_3$ and $\mathcal{D}_4$, $\mathcal{D}_5$ and $\mathcal{D}_6$, and $\mathcal{D}_7$ and $\mathcal{D}_8$, and immediately

decompose the head symbols in the unification pairs. Thereby we obtain the following four clauses consisting of exactly one unification constraint.

$$\mathcal{C}_1 : [p = \lambda x.\, x]^F \quad \mathcal{C}_2 : [p = \lambda x.\, \neg x]^F \quad \mathcal{C}_3 : [p = \lambda x.\, \bot]^F$$

$$\mathcal{C}_4 : [p = \lambda x.\, \top]^F$$

Whereas none of these unification constraints is solvable taken alone (even not by E-unification), it is possible in calculus $\mathcal{ER}$ to refute the assumption that these unification constraints are simultaneously not solvable. Like in the previous example the idea of the following derivation is to show that always one of these unification constraints must be solvable even though one cannot specify which one. The proof presented here has been automatically generated by the prototypical higher-order theorem prover LEO (Benzmüller and Kohlhase 1998b) (which implements calculus $\mathcal{ER}$) within 25 seconds on a Pentium II with 400MHz. Each line presented below introduces a new clause (the line numbering thereby corresponds to the clause numbering) by applying the specified calculus rules to previously derived clauses. For instance, line 32 describes that clause $\mathcal{C}_{32}$ is derived from clauses $\mathcal{C}_{17}$ and $\mathcal{C}_{16}$ by resolution with rule Res and immediate elimination of trivial unification constraints with rule Triv. In the proof below $s^1, \ldots, s^4$ are new Skolem constants of Boolean type introduced by the functional extensionality rule Func at the very beginning of the refutation.

5 :	Func($\mathcal{C}_4$)	$\mathcal{C}_5 : [(p\ s^3) = \top]^F$
6 :	Func($\mathcal{C}_3$)	$\mathcal{C}_6 : [(p\ s^2) = \bot]^F$
7 :	Func($\mathcal{C}_2$)	$\mathcal{C}_7 : [(p\ s^4) = (\neg\ s^4)]^F$
8 :	Func($\mathcal{C}_1$)	$\mathcal{C}_8 : [(p\ s^1) = s^1)]^F$
10 :	Equiv+Cnf($\mathcal{C}_5$)	$\mathcal{C}_{10} : [(p\ s^3)]^F$
13 :	Equiv+Cnf($\mathcal{C}_6$)	$\mathcal{C}_{13} : [(p\ s^2)]^T$
16 :	Equiv+Cnf($\mathcal{C}_7$)	$\mathcal{C}_{16} : [s^4]^T \vee [(p\ s^4)]^F$
17 :	Equiv+Cnf($\mathcal{C}_7$)	$\mathcal{C}_{17} : [(p\ s^4)]^T \vee [s^4]^F$
20 :	Equiv+Cnf($\mathcal{C}_8$)	$\mathcal{C}_{20} : [(p\ s^1)]^F \vee [s^1]^F$
21 :	Equiv+Cnf($\mathcal{C}_8$)	$\mathcal{C}_{21} : [s^1]^T \vee [(p\ s^1)]^T$
32 :	Res+Triv($\mathcal{C}_{17}$; $\mathcal{C}_{16}$)	$\mathcal{C}_{32} : [(p\ s^4)]^T \vee [(p\ s^4)]^F$
36 :	Res($\mathcal{C}_{20}$; $\mathcal{C}_{17}$)	$\mathcal{C}_{36} : [s^4]^F \vee [s^1]^F \vee [(p\ s^1) = (p\ s^4)]^F$
42 :	Dec($\mathcal{C}_{36}$)	$\mathcal{C}_{42} : [s^1]^F \vee [s^4]^F \vee [s^1 = s^4]^F$
56 :	Equiv+Cnf($\mathcal{C}_{42}$)	$\mathcal{C}_{56} : [s^1]^F \vee [s^4]^F$

76 :	Res($\mathcal{C}_{32}$; $\mathcal{C}_{21}$)	$\mathcal{C}_{76} : [s^1]^T \vee [(p\ s^4)]^T \vee [(p\ s^4) = (p\ s^1)]^F$
85 :	Dec($\mathcal{C}_{76}$)	$\mathcal{C}_{85} : [(p\ s^4)]^T \vee [s^1]^T \vee [s^4 = s^1]^F$
134 :	Equiv+Cnf($\mathcal{C}_{85}$)	$\mathcal{C}_{134} : [(p\ s^4)]^T \vee [s^1]^T \vee [s^4]^T$
141 :	Res+Triv($\mathcal{C}_{56}$; $\mathcal{C}_{16}$)	$\mathcal{C}_{141} : [(p\ s^4)]^F \vee [s^1]^F$
144 :	Res+Triv($\mathcal{C}_{56}$; $\mathcal{C}_{21}$)	$\mathcal{C}_{144} : [(p\ s^1)]^T \vee [s^4]^F$
163 :	Res+Triv($\mathcal{C}_{141}$; $\mathcal{C}_{21}$)	$\mathcal{C}_{163} : [(p\ s^1)]^T \vee [(p\ s^4)]^F$
211 :	Res($\mathcal{C}_{163}$; $\mathcal{C}_{13}$)	$\mathcal{C}_{211} : [(p\ s^1)]^T \vee [(p\ s^4) = (p\ s^2)]^F$
237 :	Dec($\mathcal{C}_{211}$)	$\mathcal{C}_{237} : [(p\ s^1)]^T \vee [s^4 = s^2]^F$
250 :	Res+Triv($\mathcal{C}_{134}$; $\mathcal{C}_{16}$)	$\mathcal{C}_{250} : [s^4]^T \vee [s^1]^T$
255 :	Res+Triv($\mathcal{C}_{134}$; $\mathcal{C}_{17}$)	$\mathcal{C}_{255} : [s^1]^T \vee [(p\ s^4)]^T$
387 :	Res+Triv($\mathcal{C}_{255}$; $\mathcal{C}_{20}$)	$\mathcal{C}_{387} : [(p\ s^4)]^T \vee [(p\ s^1)]^F$
458 :	Res($\mathcal{C}_{387}$; $\mathcal{C}_{10}$)	$\mathcal{C}_{458} : [(p\ s^1)]^F \vee [(p\ s^4) = (p\ s^3)]^F$
459 :	Res($\mathcal{C}_{387}$; $\mathcal{C}_{13}$)	$\mathcal{C}_{459} : [(p\ s^4)]^T \vee [(p\ s^1) = (p\ s^2)]^F$
492 :	Dec($\mathcal{C}_{458}$)	$\mathcal{C}_{492} : [(p\ s^1)]^F \vee [s^4 = s^3]^F$
493 :	Dec($\mathcal{C}_{459}$)	$\mathcal{C}_{493} : [(p\ s^4)]^T \vee [s^1 = s^2]^F$
519 :	Equiv+Cnf($\mathcal{C}_{493}$)	$\mathcal{C}_{519} : [(p\ s^4)]^T \vee [s^1]^F \vee [s^2]^F$
523 :	Equiv+Cnf($\mathcal{C}_{492}$)	$\mathcal{C}_{523} : [(p\ s^1)]^F \vee [s^4]^F \vee [s^3]^F$
558 :	Res+Triv($\mathcal{C}_{519}$; $\mathcal{C}_{141}$)	$\mathcal{C}_{558} : [s^2]^F \vee [s^1]^F$
592 :	Res+Triv($\mathcal{C}_{558}$; $\mathcal{C}_{21}$)	$\mathcal{C}_{592} : [(p\ s^1)]^T \vee [s^2]^F$
610 :	Res+Triv($\mathcal{C}_{558}$; $\mathcal{C}_{250}$)	$\mathcal{C}_{610} : [s^4]^T \vee [s^2]^F$
664 :	Res($\mathcal{C}_{592}$; $\mathcal{C}_{10}$)	$\mathcal{C}_{664} : [s^2]^F \vee [(p\ s^1) = (p\ s^3)]^F$
706 :	Dec($\mathcal{C}_{664}$)	$\mathcal{C}_{706} : [s^2]^F \vee [s^1 = s^3]^F$
783 :	Res+Triv($\mathcal{C}_{523}$; $\mathcal{C}_{144}$)	$\mathcal{C}_{783} : [s^3]^F \vee [s^4]^F$
820 :	Res+Triv($\mathcal{C}_{783}$; $\mathcal{C}_{610}$)	$\mathcal{C}_{820} : [s^2]^F \vee [s^3]^F$
824 :	Res+Triv($\mathcal{C}_{783}$; $\mathcal{C}_{16}$)	$\mathcal{C}_{824} : [(p\ s^4)]^F \vee [s^3]^F$
912 :	Res($\mathcal{C}_{824}$; $\mathcal{C}_{13}$)	$\mathcal{C}_{912} : [s^3]^F \vee [(p\ s^4) = (p\ s^2)]^F$
952 :	Dec($\mathcal{C}_{912}$)	$\mathcal{C}_{952} : [s^3]^F \vee [s^4 = s^2]^F$
1078 :	Equiv+Cnf($\mathcal{C}_{952}$)	$\mathcal{C}_{1078} : [s^2]^T \vee [s^4]^T \vee [s^3]^F$
1144 :	Res+Triv($\mathcal{C}_{1078}$; $\mathcal{C}_{783}$)	$\mathcal{C}_{1144} : [s^2]^T \vee [s^3]^F$
1218 :	Res+Triv($\mathcal{C}_{1144}$; $\mathcal{C}_{820}$)	$\mathcal{C}_{1218} : [s^3]^F$
1302 :	Equiv+Cnf($\mathcal{C}_{706}$)	$\mathcal{C}_{1302} : [s^3]^T \vee [s^1]^T \vee [s^2]^F$

1363 :	Res+Triv($\mathcal{C}_{1302}$; $\mathcal{C}_{558}$)	$\mathcal{C}_{1363} : [s^3]^T \vee [s^2]^F$
1377 :	Res+Triv($\mathcal{C}_{1363}$; $\mathcal{C}_{1218}$)	$\mathcal{C}_{1377} : [s^2]^F$
1454 :	Equiv+Cnf($\mathcal{C}_{237}$)	$\mathcal{C}_{1454} : [(p\ s^1)]^T \vee [s^2]^T \vee [s^4]^T$
1502 :	Res+Triv($\mathcal{C}_{1454}$; $\mathcal{C}_{144}$)	$\mathcal{C}_{1502} : [s^2]^T \vee [(p\ s^1)]^T$
1521 :	Res+Triv($\mathcal{C}_{1502}$; $\mathcal{C}_{1377}$)	$\mathcal{C}_{1521} : [(p\ s^1)]^T$
1560 :	Res($\mathcal{C}_{1521}$; $\mathcal{C}_{10}$)	$\mathcal{C}_{1560} : [(p\ s^1) = (p\ s^3)]^F$
1565 :	Res+Triv($\mathcal{C}_{1521}$; $\mathcal{C}_{20}$)	$\mathcal{C}_{1565} : [s^1]^F$
1576 :	Dec($\mathcal{C}_{1560}$)	$\mathcal{C}_{1576} : [s^1 = s^3]^F$
1643 :	Equiv+Cnf($\mathcal{C}_{1576}$)	$\mathcal{C}_{1643} : [s^3]^T \vee [s^1]^T$
1646 :	Res+Triv($\mathcal{C}_{1643}$; $\mathcal{C}_{1218}$)	$\mathcal{C}_{1646} : [s^1]^T$
1655 :	Res+Triv($\mathcal{C}_{1646}$; $\mathcal{C}_{1565}$)	$\mathcal{C}_{1655} : \square$

4.5. *Additional Examples and Case Studies*

Benzmüller (1999a) discusses several additional examples that require full extensionality reasoning – such as the following example on sets:

$$\wp(\emptyset) = \{\emptyset\}$$

It furthermore reports on case studies with the higher-order theorem prover LEO (Benzmüller and Kohlhase 1998) that demonstrate the feasibility of calculus $\mathcal{ER}$ in practice.

5. RELATED WORK

Related to calculus $\mathcal{CR}$ is the higher-order resolution approach of Jensen and Pietrzykowski (1972, 1976) which also employs a higher-order unification algorithm in order to guide the proof search. The undecidability problem of higher-order unification is thereby tackled by *dove-tailing* the generation of resolvents. Like $\mathcal{CR}$ this approach requires the extensionality axioms in the search space to ensure Henkin completeness.

Kohlhase (1994) presents a sorted variant of Huet's constrained resolution approach. Kohlhase (1995) discusses a higher-order tableaux calculus that is quite closely related to calculus $\mathcal{ER}$, as it already introduces additional calculus rules in order to improve its extensionality treatment. As is illustrated in detail in Benzmüller (1999a) the presented extensionality rules are unfortunately not sufficient to completely avoid additional extensionality axioms. The first sufficient set of extensionality rules in this sense

is presented in Benzmüller (1997), which introduces a variant of calculus $\mathcal{ER}$ as presented here.

The *theorem proving modulo* approach described in Dowek et al. (1998) is a way to remove computational arguments from proofs by reasoning modulo a congruence on propositions that is handled via rewrite rules and equations. In their paper the authors present a higher-order logic as a theory modulo.

Equality is usually treated as a defined notion in approaches and systems for automated higher-order theorem proving. This is probably the main reason why the problem of mechanising primitive equality in higher-order logic while preserving Henkin completeness has rarely been addressed in literature so far. Approaches to integrate primitive equality in a Henkin complete higher-order theorem proving approach are discussed in Snyder and Lynch (1991), Benzmüller (1999a, b). Of course, the field of higher-order term rewriting and narrowing (Prehofer 1998; Nipkow and Prehofer 1998; Nipkow 1995) is very active. But calculi developed in this context typically only address functional extensionality and do not focus on the subtle interplay between functional and Boolean extensionality that is required in a Henkin complete theorem proving approach.

The most powerful automated higher-order theorem prover currently available is (to the best knowledge of the author) the TPS-system (Andrews 1996) which employs the mating method (Andrews 1976) as inference mechanism. TPS employs a clever extensionality pre-processing mechanism which transforms embedded equations in input formulas into more appropriate ones in order to avoid later applications of the extensionality axioms. However, this does not provide a general solution and many theorems requiring non-trivial extensionality reasoning, such as Examples 3.4 and 4.4, cannot be proven this way.

6. CONCLUSION

In this paper we investigated four approaches to resolution based higher-order theorem proving: Andrews' higher-order resolution approach $\mathcal{R}$, Huet's constrained resolution approach $\mathcal{CR}$, higher-order E-resolution $\mathcal{CRE}$, and extensional higher-order resolution $\mathcal{ER}$. Thereby we focused on the extensionality treatment of these approaches and pointed to the crucial role of full extensionality for ensuring Henkin completeness. The investigated examples demonstrate that simply adding (infinitely many) extensionality axioms to the search space – as suggested for $\mathcal{R}$ and $\mathcal{CR}$ – increases the amount of blind search and is thus rather infeasible in practice.

Whereas higher-order E-unification and E-resolution indeed improves the situation in particular contexts, it does still not provide a general solution.

Calculus $\mathcal{ER}$ is the sole studied approach that can completely avoid the extensionality axioms. It's extensionality treatment is based on goal directed extensionality rules which closely connect the overall refutation search with unification by allowing for mutual recursive calls. This suitably extends the higher-order E-unification and E-resolution idea, as it turns the unification mechanism into a most general, dynamic theory unification mechanism. Unification may now itself employ a Henkin complete higher-order theorem prover as a subordinated reasoning system and the considered theory (which is defined by the sum of all clauses in the actual search space) dynamically changes. Due to the close connection of unification and refutation search it is even possible in $\mathcal{ER}$ to realise a kind of *non-constructive* reasoning on E-unifiability, as was demonstrated in this paper.

ACKNOWLEDGEMENTS

I want to thank Volker Sorge and the anonymous referee of this paper for their useful comments and contributions. I am also grateful to Michael Kohlhase for many fruitful discussions on extensional higher-order theorem proving.

NOTES

1. Conjunction elimination is provided by the rules $\vee_l^F$ and $\vee_r^F$. We note that conjunction is defined with the help of disjunction and negation; cf. Section 2.1.
2. Existential elimination is realised by the rule Π^F. For this note that existential quantification is defined with the help of universal quantification (and universal quantification with the help of Π); cf. Section 2.1.
3. It is still an open problem whether it is possible to restrict the required instances of the functional extensionality axioms in dependence of a given proof problem.
4. One may choose a bound on the allowed number of nested branchings in the search tree with rule FlexRigid.
5. Since $\mathcal{D}_o$ contains two elements, $\mathcal{D}_{o\to o}$ contains in each Henkin model at most four elements. And because of the requirement, that the function domains in Henkin models must be rich enough such that every term has a denotation, it follows that $\mathcal{D}_{o\to o}$ contains exactly the pairwise distinct denotations of the four presented function terms.

REFERENCES

Andrews, P, B.: 1971, 'Resolution in Type Theory', *Journal of Symbolic Logic* **36**, 414–432.

Andrews, P. B.: 1972, 'General Models and Extensionality', *Journal of Symbolic Logic* **37**, 395–397.

Andrews, P. B.: 1973, *Letter to Roger Hindley* dated January 22.

Andrews, P. B.: 'Refutations by Matings', *IEEE Transactions on Computers* **C-25**, 801–807.

Andrews, P. B.: 1989, 'On Connections and Higher Order Logic', *Journal of Automated Reasoning* **5**, 257–291.

Andrews, P. B., Bishop, M., Issar, S., Nesmith, D., Pfenning, F., and Xi, H.: 1996, 'TPS: A Theorem Proving System for Classical Type Theory', *Journal of Automated Reasoning* **16**, 321–353.

Barendregt, H. P.: 1984, *The Lambda Calculus – Its Syntax and Semantics*, Studies in Logic and the Foundations of Mathematics 103, Amsterdam.

Benzmüller, C.: 1997, *A calculus and a System Architecture for Extensional Higher-order Resolution*, Research Report 97-198, Department of Mathematical Sciences, Carnegie Mellon University.

Benzmüller, C.: 1999a, *Equality and Extensionality in Automated Higher-Order Theorem Proving*, Ph.D. thesis, Technische Fakultät, Universität des Saarlandes.

Benzmüller, C.: 1999b, in H. Ganzinger (ed.), *Proceedings of the 16th Conference on Automated Deduction*, Lecture Notes in Artificial Intelligence 1632, pp. 399–413, Springer.

Benzmüller, C. and Kohlhase, M.: 1997, 'Model Existence for Higher-Order Logic', SEKI-Report SR-97-09, Fachbereich Informatik, Universität des Saarlandes.

Benzmüller, C. and Kohlhase, M.: 1998a, 'Extensional Higher-order Resolution', in Kirchner and Kirchner (eds.), *Proceedings of the 15th Conference on Automated Deduction*, Lecture Notes in Artificial Intelligence 1421, pp.56–72, Springer.

Benzmüller, C. and Kohlhase, M. 1998b, 'LEO – A Higher-order Theorem Prover, in Kirchner and Kirchner (eds.), *Proceedings of the 15th Conference on Automated Deduction*, Lecture Notes in Artificial Intelligence 1421, pp. 139–144, Springer.

Baader, F. and Siekmann, J.: 1994, 'Unification Theory', in D. M. Gabbay, C. J. Hogger, J. A. Robinson (eds.), *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 2: Deduction Methodologies*, Oxford, Chapter 2.2

Church, A.: 1940, 'A formulation of the Simple Theory of Types', *Journal of Symbolic Logic* **5**, 56–68.

Dowek, G., Hardin, T., and Kirchner, C.: 1998, *Theorem Proving Modulo, Rapport de Recherche 3400*, Institut National de Recherche en Informatique et en Automatique.

Dougherty, D. and Johann, P.: 1992, 'A Combinatory Logic Approach to Higher-order E-unification', in D. Kapur (ed.), *Proceedings of the 11th Conference on Automated Deduction*, Lecture Notes in Artificial Intelligence 607, pp. 79–93, Springer.

Goldfarb, W. D. 1981, 'The Undecidability of the Second-order Unification Problem', *Theoretical Computer Science* **13**, 225–230.

Henkin, L.: 1950, 'Completeness in the Theory of Types', *Journal of Symbolic Logic* **15**, 81–91.

Huet, G. P.: 1972, *Constrained Resolution: A Complete Method for Higher Order Logic*, Ph.D. thesis, Case Western Reserve University.

Huet, G. P.: 1973, 'A Mechanization of Type Theory', in D. E. Walker and L. Norton (eds.), *Proceedings of the 3rd International Joint Conference on Artificial Intelligence*, pp. 139–146.

Huet, G. P.: 1973, 'The Undecidability of Unification in Third Order Logic', *Information and Control* **22**, 257–267.

Huet, G. P.: 1975, 'A Unification Algorithm for Typed λ-calculus', *Theoretical Computer Science* **1**, 27–57.

Jensen, D. C. and Pietrzykowski, T.: 1972, 'A Complete Mechanization of ω-order Type Theory', in *Proceedings of the ACM annual Conference*, volume 1, 89–92.

Jensen, D. C. and Pietrzykowski, T.: 1976, 'Mechanizing ω-order Type Theory through Unification', *Theoretical Computer Science* **3**, 123–171.

Kohlhase, M.: 1994, *A Mechanization of Sorted Higher-Order Logic Based on the Resolution Principle*, Ph.D. thesis, Fachbereich Informatik, Universität des Saarlandes.

Kohlhase, M.: 1995, 'Higher-Order Tableaux', in P. Baumgartner, R. Hähnle, and J. Posegga (eds.), *Theorem Proving with Analytic Tableaux and Related Methods*, Lecture Notes in Artificial Intelligence 918, pp. 294–309, Springer.

Lucchesi, C. L.: 1972, *The Undecidability of the Unification Problem for Third Order Languages*, Report CSRR 2059, University of Waterloo, Waterloo, Canada.

Miller, D.: 1983, *Proofs in Higher-Order Logic*, Ph.D. thesis, Carnegie Mellon University.

Nipkow, T.: 1995, 'Higher-order Rewrite Systems', in J. Hsiang (ed.), *Rewriting Techniques and Applications*, *6th International Conference*, Lecture Notes in Computer Science 914, Springer.

Nipkow, T. and Prehofer, C.: 1998, 'Higher-order Rewriting and Equational Reasoning', in W. Bibel and P. Schmitt (eds.), *Automated Deduction – A Basis for Applications*, Dordrecht, Applied Logic Series, pp. 399–430.

Nipkow, T. and Qian, Z,: 1991, 'Modular Higher-order E-unification', in R. V. Book (ed.), *Proceedings of the 4th International Conference on Rewriting Techniques and Applications*, Lecture Notes in Artificial Intelligence 488, pp. 200–214, Springer.

Prehofer, C.: 1998, *Solving Higher-Order Equations: From Logic to Programming*, Progress in theoretical computer science, Birkhäuser.

Qian, Z. and Wang K.: 1996, 'Modular Higher-order Equational Preunification', *Journal of Symbolic Computation* **22**, 401–424.

Robinson, J. A.: 1965, 'A Machine-oriented Logic Based on the Resolution Principle', *Journal of the Association for Computing Machinery* **12**, 23–41.

Siekmann, J. H.: 1989, 'Unification Theory', *Journal of Symbolic Computation* **7**, 207–274.

Smullyan, R. M. 1963, 'A Unifying Principle for Quantification Theory', *Proceedings of the National Acadamy of Sciences, USA 49*, pp. 828–832.

Snyder, W.: 1990, 'Higher Order E-unification', in M. Stickel (ed.), *Proceedings of the 10th Conference on Automated Deduction*, Lecture Notes in Artificial Intelligence 449, pp. 573–578, Springer.

Snyder, W. and Gallier, J.: 1989, 'Higher-order Unification Revisited: Complete Sets of Transformations', *Journal of Symbolic Computation* **8**, 101–140.

Snyder, W. and Lynch, C.: 1991, 'Goal-directed Strategies for Paramodulation', in R. V. Book (ed.), *Proceedings of the 4th International Conference on Rewriting Techniques and Applications*, Lecture Notes in Artificial Intelligence 488, pp. 200–214, Springer.

Wolfram, D. A.: 1993, *The Clausal Theory of Types*, Cambridge, Cambridge Tracts in Theoretical Computer Science 21.

Fachbereich Informatik,
Universität des Saarlandes
D-66041 Saarbrücken
Germany
E-mail: chris@ags.uni-sb.de