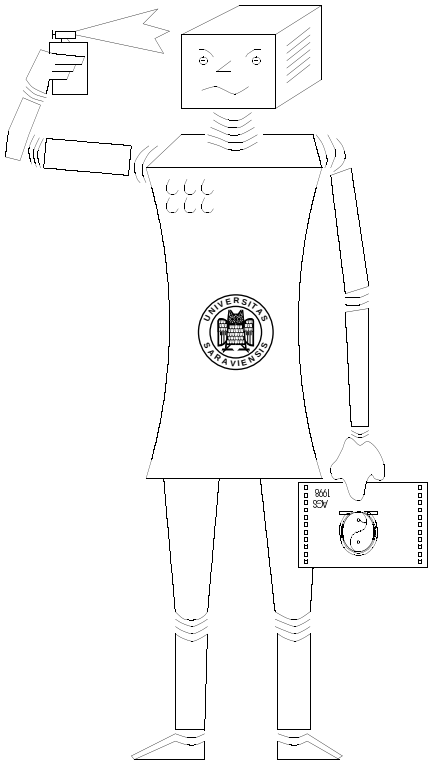
UNIVERSITÄT DES SAARLANDES
FACHBEREICH INFORMATIK
D-66041 SAARBRÜCKEN
GERMANY
WWW: http://www.ags.uni-sb.de/

SEKI Report



An Adaptation of Paramodulation and RUE-Resolution to Higher-Order Logic

Christoph Benzmüller
chris@cs.uni-sb.de

SEKI Report SR-98-07

Abstract

This techreport presents two approaches to primitive equality treatment in higher-order (HO) automated theorem proving: a calculus $\mathcal{EP}$ adapting traditional first-order (FO) paramodulation [RW69] , and a calculus $\mathcal{ERUE}$ adapting FO RUE-Resolution [Dig79] to HO logic (based on Church's simply typed λ-calculus). $\mathcal{EP}$ and $\mathcal{ERUE}$ extend the extensional HO resolution approach $\mathcal{ER}$ [BK98a]. In order to reach Henkin completeness without the need for additional extensionality axioms both calculi employ new, positive extensionality rules analogously to the respective negative ones provided by $\mathcal{ER}$ that operate on unification constraints. As the extensionality rules have an intrinsic and unavoidable difference-reducing character the HO paramodulation approach looses its pure term-rewriting character. On the other hand examples demonstrate that the extensionality rules harmonise rather well with the difference-reducing HO RUE-resolution idea.

Contents

1 Introduction **2**

2 Higher-Order Logic **4**

3 Extensional Higher-Order Resolution: $\mathcal{ER}$ **7**
3.1 A Review of $\mathcal{HORES}$ and $\mathcal{ER}$ 7
3.2 Basic Definitions 12
3.3 Lifting Properties 15
3.4 Completeness 16
3.5 Theorem Equivalence 18

4 Primitive Equality **20**

5 Extensional Higher-Order Paramodulation: $\mathcal{EP}$ **21**
5.1 A Naive and Incomplete Adaption of Paramodulation 21
5.2 Positive Extensionality Rules 25
5.3 Basic Definitions 25
5.4 Lifting Properties 26
5.5 Completeness 27
5.6 Theorem Equivalence 33

6 Extensional Higher-Order RUE-Resolution: $\mathcal{ERUE}$ **33**
6.1 Resolution on Unification Constraints 33
6.2 Basic Definitions 34
6.3 Lifting Properties 35
6.4 Completeness 35
6.5 Theorem Equivalence 40

7 Examples **40**
7.1 Decomposition in $\mathcal{ER}$ 40
7.2 Leibniz Equality and Alternative Definitions in $\mathcal{ER}$ 42
7.3 Positive Extensionality Rules in $\mathcal{EP}$ and $\mathcal{ERUE}$ 43
7.4 Comparing $\mathcal{EP}$ and $\mathcal{ERUE}$ 45

8 Conclusion **50**

1 Introduction

The efficient and adequate mechanisation of the equality relation is one of the most challenging questions in automated theorem proving. Within the context of first-order automated theorem proving many different techniques and calculi have been developed after Robinson came up with his resolution calculus in 1965 [Rob65] and a few years later with the paramodulation calculus [RW69]. They can be roughly divided into to *term-rewriting approaches* and *difference-reducing approaches*. Typical representatives for the former class are paramodulation [RW69] and all its refinements such as the state of the art superposition calculi [BGLS92]. A typical representative for the latter class is RUE-resolution [Dig79] which can be seen as a generalisation of E-resolution [Dar68]. If one considers the results of the first-order, term-rewriting based theorem provers such as SPASS [Wei97] or WALDMEISTER [HBVL97] at the the CADE–14 and CADE–15 system competitions it seems that recently in first-order context the term-rewriting approaches have taken the lead over the difference-reducing ones – if one ignores the admittedly important field of induction based theorem proving.

For higher-order logic only a few calculi have been discussed in literature so far and the most famous approaches are the resolution calculi by Andrews [And71], Jensen and Pietrzykowski [JP72], and [Hue72, Hue73a]. And in contrast to the first-order context the available approaches for an efficient and suitable equality treatment in higher-order theorem proving are rather limited. Sure, there are interesting developments in the field of higher-order term-rewriting and many publications, e.g. [NR95, JR96, JR98, LB98, Pre98, NQ92], concentrate on the development of well-founded term orderings for (fragments of) higher-order languages which is the basic prerequisite for the adaptation of first-order term-rewriting approaches to the higher-order setting. But unfortunately rather few discuss the applicability of term-rewriting techniques within higher-order theorem proving sufficiently.[1]

In this paper we motivate why an efficient and suitable integration of a higher-order term-rewriting approach into a Henkin complete theorem prover is very challenging – aside from the well-founded ordering problem. Concretely we define the calculus $\mathcal{EP}$ for *extensional higher-order paramodulation* that extends the extensional higher-order resolution calculus $\mathcal{ER}$ [BK98a] for classical higher-order logic based on Church simply typed λ-calculus. We show that the naive and straightforward adaption of first-order paramodulation to the higher-order setting does not automatically lead to calculus that is complete with respect to Henkin models[2] — although Henkin completeness without the need for additional axioms has been shown in [BK98a] for the underlying extensional higher-order resolution calculus $\mathcal{ER}$. The new problems are (again) caused by the extensionality principles which express that two functions are equal iff they are equal on all arguments (functional extensionality) and that on booleans the equality relation coincides with the equivalence relation (boolean extensionality). Sure, the problems dissappear as soon as one adds respective extensionality axioms to the calculus as it is, for instance, necessary in the higher-order resolution approaches [And71, JP72, Hue72, Hue73a] or [Wol94]. But this is obviously awkward to manage in practice as the extensionality axioms heavily increase the search space and as generally infinitely many functional extensionality axioms are needed (note that there are infinitely many functional types). Another solution – as for instance suggested in [Wol93] – is to employ higher-order E-unification. In this case we have to add the extensionality principles to the theory E, which means that we are employing extensional higher-order E-unification. Now consider the following extensional higher-order unification problem defined by the two set descriptions $\lambda X_{nat}.\ (X < 10) \wedge (\exists Y_{nat}.\ (Y \times 2) = X)$ and $\lambda X_{nat}.\ (\exists Y_{nat}.\ (X = (Y \times 2)) \wedge (X < 10)$. It becomes obvious that a Henkin complete higher-order theorem prover needs to be recursively employed within extensional higher-order unification or extensional higher-order E-unification.

[1] In higher-order theorem we are interested especially in examples like $((a_o \wedge b_o) = c_o \wedge (p_{o\to o}\ (b \wedge a)) \Rightarrow p\ c$. A term-rewriting approach that employs Pure syntactical matching or unification is obviously to weak in order to find the trivial refutation for this example.)

[2] Whereas there can no complete calculi with respect to the intuitive standard semantics, it is well known by Henkin [Hen50] that there are complete calculi if one considers a weaker notion of semantics, that only presupposes that the respective function domains are subsets of the sets of all functions belonging to a certain functional type. Henkin semantics is the most general notion that is known to allow complete calculi.

Higher-order E-unification approaches are discussed in [Sny90, Sny91, NQ91, Nip93, MW94]; but from the perspective of automated higher-order theorem proving these papers do not sufficiently tackle the problem of extensionality.

The extensional higher-order resolution calculus $\mathcal{ER}$ introduced in [BK98a], which works within a higher-order language that omits a primitive notion of equality and treats equality as defined concept instead (by Leibniz' definition or any other definition principle), solves the extensionality problem for a higher-order language without primitive equality by adding special extensionality rules to the unification rules. They turn the embedded higher-order (pre-)unification calculus due to the ideas of [Hue75] into a corresponding test calculus for extensional higher-order (pre-)unification or even for extensional higher-order E-(pre-)unification if we add the equations defining (the axiomatisable) theory E as positive unit equations into the search space. This is nicely illustrated by the example discussed in subsection 7.1. The new extensionality rules operate on unification constraints and it is not surprising that they realise a close interleaving of the unification process and the overall refutation search by allowing for recursive calls of the overall refutation process from within the unification process.

As we will discuss in this paper the extensionality treatment on unification constraints turns out to be not sufficient if one additionally adds primitive equality symbols to the higher-order language. Even if we add a paramodulation (and reflexivity) rule to the extensional higher-order resolution approach the extended approach still lacks Henkin completeness. The reason is that new problem with the extensionality principles arises as in contrast to the first-order case isolated positive equations can be contradictory by themselves in higher-order logic (consider for instance the unit clause $[a_o = \neg a_o]^T$ or the examples discussed in subsection 7.3). In order to obtain a Henkin complete approach again additional extensionality rules are needed in our extensional higher-order paramodulation approach $\mathcal{EP}$. In contrast to the negative extensionality rules for unification constraints in the underlying calculus $\mathcal{ER}$, they operate on positive equation literals.

Unfortunately the new positive extensionality rules have (which is not surprising) an intrinsic difference reducing character such that the question arises if and how they can fruitfully be employed in a calculus with an overall term-rewriting idea. We want to point out that this not only affects the adaptation of first-order paramodulation to higher-order logic but also the potential adaption of all refinements of paramodulation such as the superposition calculus [BGLS92].

As an opposing candidate to extensional higher-order paramodulation $\mathcal{EP}$ we therefore also adapt the first-order RUE-resolution approach [Dig79] into the *extensional higher-order RUE-resolution* calculus $\mathcal{ERUE}$. Analogously to the paramodulation case we can proof Henkin completeness only if we add the new positive extensionality rules. But the RUE-resolution approach and the new extensionality rules share a difference-reducing character such that at least from an abstract point of view they may harmonise much better as the paramodulation and positive extensionality rules.

In this paper we discuss Henkin completeness for all three approaches $\mathcal{ER}$, $\mathcal{EP}$ and $\mathcal{ERUE}$. One aim thereby is to illustrate the similarities and differences between the three approaches and to present the completeness proofs in a uniform way. On the one hand these proofs are oriented on the ideas of the corresponding completeness proof already presented for extensional higher-order resolution calculus $\mathcal{ER}$ in [BK98a]. On the other hand we here want to get rid of the rather unintuitive clause isomorphism within the complicate lifting lemma employed in [BK98a] or [Koh94] which are susceptible to errors. This can be achieved by adding the well known *FlexFlex*-unification rule to the calculus. Thus, the Henkin completeness proofs presented in this paper take this additional rule into account and thereby gains clarity and simplicity — especially within the lifting lemmata. In practice one certainly wants to omit this infinitely branching rule which guesses instantiations for flexible heads of unification pairs. The fact that this rule is unsuitable in practice was also a main motivation for Huet's important contribution in [Hue75] where he discusses higher-order pre-unification, i.e. a higher-order unification test that avoids rule *FlexFlex*. Higher-order pre-unification is known to be sufficient within a refutation approach, e.g. for Huet's traditional higher-order resolution [Hue72, Hue73a] as one is only interested in finding one instantiation of the given clause set that makes it contradictory.

It is thus clear that we want to get rid of the *FlexFlex* rule in our approach as well. But

unfortunately the author was up to now not able to carry out a formal proof for the admissibility of rule *FlexFlex* in either one of the approaches $\mathcal{ER}$, $\mathcal{EP}$ or $\mathcal{ERUE}$. Consequently Henkin completeness of all three approaches (without rule *FlexFlex*) is presented only as a conjecture which gains evidence by the examples examined in this paper as well as by the case studies carried out with the LEO-prover for extensional higher-order resolution (and meanwhile for the new paramodulation or RUE-resolution approaches). It is not surprising that for the proof of the admissibility claim for rule *FlexFlex*, which the author expects to be analogous for all three approaches, a rather complicate proof transformation argument will be needed. This is due to the fact that different to Huet's traditional higher-order resolution approach eager unification becomes essential within our extensional higher-order resolution (paramodulation or RUE-resolution) approach is nicely illustrated by example $\mathbf{E}_3$ in [BK98a] or $\mathbf{E}^{Dec}$ in subsection 7.1 in this paper.

All completeness proofs are presented in a uniform way and most of the needed lemmata are analogous or even identical. For our completeness proofs we employ the abstract consistency method which was first introduced by Smullyan [Smu63] for first-order logic and later extended to higher-order logic by Peter Andrews [And71]. Unfortunately, the latter does not allow completeness proofs with respect to Henkin semantics but only for a weaker notion of semantics. In our proofs we therefore employ the abstract consistency method discussed in [BK97b] which further extends [And71] and which ensures completeness with respect to Henkin semantics (even with primitive equality).

The paper is organised as follows: The basic concepts are introduced in section 1 and the extensional higher-order resolution calculus $\mathcal{ER}$ is reviewed in section 3. We briefly discuss some important aspects of defined and primitive equality in higher-order logic in section 4. The extensional higher-order paramodulation calculus $\mathcal{EP}$ and the extensional higher-order RUE-resolution calculus $\mathcal{ERUE}$ are then presented in sections 5 and 6. Section 7 illustrates the practical usage of all three approaches by examples and we conclude the paper in section 8.

2 Higher-Order Logic

We consider a higher-order logic based on Church's simply typed lambda calculus [Chu40] and choose the set of base types $\mathcal{BT}$ to consists of the types ι and o, where o denotes the set of truth values and ι the set of individuals. The set of all types $\mathcal{T}_{\mathcal{BT}}$ is inductively defined over $\mathcal{BT}$ and the type constructor $\rightarrow$. We assume that our signature Σ contains a countably infinite set of variables $\mathcal{V}_\tau$ and constants $\mathcal{C}_\tau$ for every type $\tau \in \mathcal{T}$. Additionally we postulate the existence of the logical connectives $\neg_{o\rightarrow o}$, $\vee_{o\rightarrow o\rightarrow o}$, $\Pi_{(\alpha\rightarrow o)\rightarrow o}$ (in short Π^α) for every type $\alpha \in \mathcal{T}$ in Σ. A signature that contains additionally the logical connectives $=_{\alpha\rightarrow\alpha\rightarrow o}$ (in short $=^\alpha$) for all types $\alpha \in \mathcal{T}$ is noted by $\Sigma^=$. All the logical logical connectives in Σ or $\Sigma^=$ denote their intuitive semantical counterparts.

The remaining logical connectives are defined as abbreviations of the given ones: $\mathbf{A} \wedge \mathbf{B} := \neg(\neg\mathbf{A} \vee \neg\mathbf{B})$, $\mathbf{A} \Rightarrow \mathbf{B} := \neg\mathbf{A} \vee \mathbf{B}$, $\mathbf{A} \Leftrightarrow \mathbf{B} := (\mathbf{A} \Rightarrow \mathbf{B}) \wedge (\mathbf{B} \Rightarrow \mathbf{A})$, $\forall X_\alpha.\ \mathbf{A}_o := \Pi_{(\alpha\rightarrow o)\rightarrow o}(\lambda X_\alpha.\ \mathbf{A})$, $\exists X_\alpha.\ \mathbf{A}_o := \neg\forall X_\alpha.\ \neg\mathbf{A}_o$. All other constants are called *parameters*, since the argumentation in this paper is parametric in their choice[3].

Unlike stated otherwise, 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 or term is either not of importance or uniquely determined by the given context we avoid its explicit mention.

We denote the set of all terms over signature Σ ($\Sigma^=$) by $\mathcal{L}_\Sigma$ ($\mathcal{L}_{\Sigma=}$). Terms of type o are also called *propositions* and closed propositions are called *sentences*. $\mathbf{A}$ is called atomic if its $\beta\eta$-normalform does not have a logical connective at head position.

In order to avoid confusion we clarify the meaning of the different equality symbols used in this paper. As already specified $=^\alpha \in \mathcal{C}_{\alpha\rightarrow\alpha\rightarrow o}$ is the syntactic equality constant in our higher-order language. We will illustrate below that equality can also be defined in higher-order logic and we

[3] In particular, we do not assume the existence of description or choice operators. For a detailed discussion of the semantic issues raised by the presence of these logical constants see [And72].

refer to this definition with $\doteq$. The intuitive semantical equality relations in $\mathbf{Dom}_{\alpha\to\alpha\to o}$ are denoted by q^α. For the meta-level argumentations in this paper we use $\equiv$ and for definitions $:=$.

To ease readability we assume right-associativity for the type constructor $\to$ and left-associativity of function application ($A_{\alpha\to\beta\to\gamma}\ B_\alpha\ C_\beta := ((A_{\alpha\to\beta\to\gamma}\ B_\alpha)\ C_\beta)$). Sometimes we abbreviate function applications by $h_{\alpha_1\to\cdots\to\alpha_n\to\beta}\ \overline{\mathbf{U}^n_{\alpha_n}}$ which stands for $(\cdots(h_{\alpha_1\to\cdots\to\alpha_n\to\beta}\ \mathbf{U}^1_{\alpha_1})\cdots \mathbf{U}^n_{\alpha_n})$. A dot "." occurring in a λ-term stands for a left bracket whose mate is as far to the right as consistent with all other brackets and the construction of the term. We allow further to avoid brackets in every case, where the construction of an expression is uniquely determined by the context.

The structural equality relation of $\mathcal{HOL}$ is induced by $\beta\eta$-reduction

$$(\lambda X.\ \mathbf{A})\ \mathbf{B} \longrightarrow_\beta [\mathbf{B}/X]\mathbf{A} \qquad\qquad (\lambda X.\ \mathbf{C}\ X) \longrightarrow_\eta \mathbf{C}$$

where X is not free in $\mathbf{C}$. It is well-known (c.f. [Bar84]), that the reduction relations β, η, and $\beta\eta$ are terminating and confluent, so that there are unique normal forms for each term $\mathbf{T}$ denoted by $\mathbf{T}_{\downarrow\beta\eta}$, $\mathbf{T}_{\downarrow\beta}$, $\mathbf{T}_{\downarrow\eta}$. Another important normal form used in this paper is the head normal form (which is unique only with respect to $\beta\eta$-equality): a term $\lambda\overline{X^n}.\ h\overline{\mathbf{U}^m}$ is in head-normal form iff h is a variable or a constant. The head-normal form of a term $\mathbf{T}$ is denoted by $\mathbf{T}_{\downarrow h}$.

The definitions of *free* and *bound* variables, *substitutions* and the *application of substitution* are as usual (see [Bar84]). In this paper we denote the set of free variables of a term $\mathbf{T}$ (analogously for literals and clauses) by $\mathbf{Free}(\mathbf{T})$. Whereas the usual application of a substitution $[\mathbf{A}/X]$ to $\mathbf{T}$ is denoted by $[\mathbf{A}/X]\mathbf{T}$, we refer with $\mathbf{T}_{[\mathbf{A}/X]}$ to the combination of usual substitution with subsequent reduction of the resulting term (literal or clause) to head normal form.

When we speak of a *Skolem term* s_α *for a clause* C and $\mathbf{Free}(\mathcal{C}) \equiv \{X^1_{\alpha^1}\cdots X^n_{\alpha^n}\}$, then s_α is an abbreviation for the term $(f^n_{\alpha^1\to\cdots\to\alpha^n\to\alpha}X^1\cdots X^n)$, where f is a new constant from $\mathcal{C}_{\alpha^1\to\cdots\to\alpha^n\to\alpha}$ and n specifies the number of necessary arguments for f. The latter is important as a naive treatment of Skolemisation results in a calculus that is not sound with respect to Henkin models, since Skolem functions are special choice functions[4], which are not guaranteed to exist in Henkin models. A solution due to [Mil83] is to associate with each Skolem constant the minimum number of arguments the constant has to be applied to. Skolemisation becomes sound, if any Skolem function f^n only occurs in a *Skolem term*, i.e. a formula $\mathbf{S} = f^n\overline{\mathbf{A}^n}$, 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.

For a general introduction to higher-order unification and especially for the definition of a set of *general bindings* $\mathcal{GB}^h_\gamma$ for a type γ and a (head-)constant h we refer to [GS89].

The calculi in this paper are defined on *clauses* which are disjunctions of *literals* (e.g., $[q_{\alpha\to o}X_\alpha]^T \vee [p_{\alpha\to o}X_\alpha]^F \vee [c_\alpha = X_\alpha]^F$). For literals we differentiate between *pre-literals* and *proper literals*. A *pre-literal* consists of a proposition $\mathbf{N}_o$ in head-normal form (*atom*) and a polarity T or F stating if this literal is positive or negative. We call a literal *proper* iff it contains no logical constant beside $=$ at head position.

We further differentiate between *positive literals*, *negative literals* and *unification constraints*. Unification constraints refer to negative literals with primitive equations as atoms. We want to point out that for the calculi discussed in this paper unification constraints are treated in one of the following ways:

- As *special negative literals* with a special head symbol $=^\alpha \notin \mathcal{C}$, i.e. $=^\alpha$ is not a logical connective. By special literal we mean that this literal is treated as a unification constraint only, which means that no rule but the unification rules are allowed to operate on them. (This will be the case in the extensional higher-order resolution $\mathcal{ER}$ discussed in section 3.)
- As a special negative literals with a logical connective $=^\alpha \in \mathcal{C}$ as head symbol. Special negative literal means that despite the fact that $=^\alpha$ is a logical connective provided by the signature no rule but the unification rules are allowed to operate on them. (This will be the case in the extensional higher-order paramodulation calculus $\mathcal{EP}$ in section 5.)

[4] They choose an existential witness from the set of possible witnesses for an existential formula.

- As ordinary negative literals with the logical connective $=^\alpha \in \mathcal{C}$ as head symbol. In this case all rules, e.g. the resolution and factorisation rules, are allowed to operate on these literals. (This holds for the extensional higher-order paramodulation calculus $\mathcal{ERUE}$ in section 6)

A clause $\mathcal{C}$ is called a *proper clause* iff it is in *clause normal form*, i.e. if all literals of $\mathcal{C}$ are proper. Otherwise we call $\mathcal{C}$ a *pre-clause*. Similar a clause $\mathcal{C}$ is in *head-normal form* iff all its literals are.

An unification constraint $U := [X_\alpha = \mathbf{N}_\alpha]^F$ or $U := [\mathbf{N}_\alpha = X_\alpha]^F$ is called *solved* iff $X_\alpha \notin \mathbf{Free}(\mathbf{N}_\alpha)$. In this case X is called the *solved variable* of U. Furthermore, a unification constraint is in *head-normal form* $[(H\ \overline{\mathbf{U}^n}) = (G\ \overline{\mathbf{V}^m})]^F$ where $n \geq 1$ is called a *flex-flex constraint* (flex-rigid constraint) iff H and G are variables (either H or G is a variable), i.e. if both hand sides of the unification constraint are in head-normal form.

Let $\mathcal{C} := L^1 \vee \cdots \vee L^n \vee U^1 \vee \cdots \vee U^m$ be a clause with unification constraints $U^1 \vee \cdots \vee U^m$ $(1 \leq m)$. Then a disjunction $U^{i_1} \vee \cdots \vee U^{i_k}$ $(i_j \in \{1, \cdots, m; 1 \leq j \leq k)$ of solved unification constraints occurring in $\mathcal{C}$ is called *solved for* $\mathcal{C}$ iff for every $U^{i_j} (1 \leq j \leq k)$ holds: the solved variable of U^{i_j} does not occur free in any of the U^{i_l} for $l \neq j; 1 \leq l \leq k$.

We define a clause $\mathcal{C}$ to be *empty* (denoted by $\Box$) iff $\mathcal{C}$ consists only of *flex-flex constraints*. As it is well known that any set of *flex-flex constraints* is unifiable we know that $\Box$ is satisfiable.

Given a calculus R, i.e. a set of rules $r_n (n \geq 0)$ defined on clauses, we define the following derivation relations: $\Phi \vdash^{r_n} \mathcal{C}$ ($\mathcal{C}' \vdash^{r_n} \mathcal{C}$) iff $\mathcal{C}$ is the result of the one step application of rule $r_n \in R$ to premise clauses $\mathcal{C}'_i \in \Phi$ (to premise clause $\mathcal{C}'$). Multiple step derivations within a calculus R, e.g. $\Phi_1 \vdash^{r_{i_1}} \ldots \vdash^{r_{i_k}} \Phi_k$ (or $\mathcal{C}_1 \vdash^{r_{i_1}} \ldots \vdash^{r_{i_k}} \mathcal{C}_k$) where $k \geq 0$, are abbreviated by $\Phi_1 \vdash_R \Phi_k$ (or $\mathcal{C}_1 \vdash_R \mathcal{C}_k$). Derivations in a calculus r of exactly n steps are symbolised by $\vdash_R^n$.

A rule r is called *admissible* in one of our resolution calculi R, iff adding rule r to R does not increase the set of refutable formulas. Furthermore, a rule r is called *derivable* in R iff each application of rule r can be replaced by an alternative derivation in R.

It is matter of folklore that equality can directly be expressed in $\mathcal{HOL}$, e.g. by the *Leibniz definition*, so that a pimitive notion of equality (expressed by a primitive constant $=$ in Σ) is not strictly needed; we will use this observation in this paper when we treat equality as a defined notion. Leibniz equality defines two terms to be equal, iff they have the same properties:

$$\doteq^\alpha := \lambda X_\alpha . \lambda Y_\alpha . \forall P_{\alpha \to o} . PX \Rightarrow PY$$

The *functional extensionality principle* says that two functions are equal iff they are equal on all arguments, which can be formulated by the following schematic λ-term: $\forall M_{\alpha\to\beta} . \forall N_{\alpha\to\beta} . (\forall X . (MX) = (NX)) \Leftrightarrow (M = N)$. This term is schematic with respect to the (arbitrary) types α and β. The *extensionality principle for truth values* states that on the set of truth values equality and equivalence relation coincide: $\forall P_o . \forall Q_o . (P = Q) \Leftrightarrow (P \Leftrightarrow Q)$.

A *standard model* for $\mathcal{HOL}$ provides a fixed set $\mathcal{D}_\iota$ of individuals, and a set $\mathcal{D}_o := \{\mathtt{T}, \mathtt{F}\}$ of truth values. All the domains for the complex types are defined inductively: $\mathcal{D}_{\alpha\to\beta}$ is the set of functions $f : \mathcal{D}_\alpha \to \mathcal{D}_\beta$. The evaluation $\mathcal{I}_\varphi$ with respect to an interpretation $\mathcal{I} : \Sigma \to \mathcal{D}$ of constants and an assignment φ of variables is obtained by the standard homomorphic construction that evaluates a λ-abstraction with a function, whose operational semantics is specified by β-reduction.

Henkin models only require that $\mathcal{D}_{\alpha\to\beta}$ has enough members that any well-formed formula can be evaluated[5]. Note that with this generalised notion of a model, there are less formulae that are valid in all models (intuitively, for any given set of formulae there are more possibilities for counter-models). Thus the generalisation to Henkin models restricts the set of valid formulae sufficiently, so that all of them can be proven by the resolution calculus presented in this paper.

In Henkin models (as well as in standard semantics) Leibniz equality denotes the intuitive equality relation and both extensionality principles are valid. For details we refer to [BK97b].

We define *satisfiability* and *validity* of a formula $\mathbf{F}$ or set of formulas Φ with respect to a model $\mathcal{M}$ as usual and we refer to he latter notion with $\mathcal{M} \models \mathbf{F}$ or $\mathcal{M} \models \Phi$.

[5] In other words: the functional universes are rich enough to satisfy the comprehension axioms.

For our completeness proofs, we will employ the abstract consistency method. This proof principle was first introduced by Smullyan [Smu63] for first-order logic and later extended to higher-order logic by Andrews [And71]. Unfortunately the latter one does not allow completeness proofs with respect to Henkin semantics but only for the rather weak semantical notion of V-complexes (see [And71]; each Henkin models is also an V-complex, but not vice a versa). Therefore we employ the abstract consistency method described in [BK97b] which extends Andrews definition such that completeness proofs with respect to Henkin semantics become possible. The main result in [BK97b] is the model existence theorem as stated below. For the proof we refer to [BK97b].

Theorem 2.1 (Henkin Model Existence). *1. Let $\Phi \subseteq \mathcal{L}_\Sigma$ be a set of closed formulas and let Γ_Σ be a saturated abstract consistency class (ACC) for Henkin models (see definition 2.2) and $\Phi \in \Gamma_\Sigma$, then there exists a Henkin model $\mathcal{M}$ such that $\mathcal{M} \models \Phi$.*

2. Let $\Phi \subseteq \mathcal{L}_{\Sigma=}$ be a set of closed formulas and Let Γ_Σ be a saturated abstract consistency class (ACC) for Henkin models with primitive equality (see definition 2.3) and $\Phi \in \Gamma_\Sigma$, then there exists a Henkin model $\mathcal{M}$ such that $\mathcal{M} \models \Phi$.

Definition 2.2 (ACC for Henkin Models). Let Σ be a signature and let Γ_Σ be a class of sets of Σ-propositions. If the following conditions hold for all $\mathbf{A}, \mathbf{B} \in cwff_o(\Sigma)$, $\mathbf{F}, \mathbf{G} \in cwff_{\alpha\to\beta}(\Sigma)$[6] and $\Phi \in \Gamma_\Sigma$ then we call Γ_Σ an **abstract consistency class for Henkin models.**

Saturated If $\mathbf{A}$ is closed then $\Phi \cup \{\mathbf{A}\} \in \Gamma_\Sigma$ 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 $\Phi * \mathbf{A} \in \Gamma_\Sigma$.

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

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

$\nabla_\wedge$ 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 $\Phi * \mathbf{FW} \in \Gamma_\Sigma$ for each $\mathbf{W} \in cwff_\alpha(\Sigma)$.

$\nabla_\exists$ If $\neg\Pi^\alpha\mathbf{F} \in \Phi$, then $\Phi * \neg(\mathbf{F}w) \in \Gamma_\Sigma$ for any constant $w \in \Sigma_\alpha$, which does not occur in Φ.

∇_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 $\Phi * \neg(\mathbf{F}w \doteq^\beta \mathbf{G}w) \in \Gamma_\Sigma$ for any constant $w \in \Sigma_\alpha$, which does not occur in Φ.

Definition 2.3 (ACC for Henkin models with primitive equality). An abstract consistency class for Henkin models Γ_Σ that fulfils ∇_e for all $\mathbf{A}, \mathbf{B} \in cwff_o(\Sigma)$, $\mathbf{F} \in cwff_{\alpha\to o}(\Sigma)$ and $\Phi \in \Gamma_\Sigma$ is called an **Abstract Consistency Class for Henkin models with primitive equality.**

∇_e (r) $\neg(\mathbf{A} =^\alpha \mathbf{A}) \notin \Phi$
(s) if $\mathbf{F}[\mathbf{A}]_p \in \Phi$ and $\mathbf{A} = \mathbf{B} \in \Phi$, then $\Phi * \mathbf{F}[\mathbf{B}]_p \in \Gamma_\Sigma$

3 Extensional Higher-Order Resolution: $\mathcal{ER}$

3.1 A Review of $\mathcal{HORES}$ and $\mathcal{ER}$

In this section we slightly modify the extensional higher-order resolution calculus $\mathcal{ER}$ as introduced in [BK98a]. This calculus is the first machine oriented higher-order refutation calculus that is complete with respect to Henkin semantics without the need for additional axioms such as the extensionality axioms. The calculus rules of $\mathcal{ER}$ can be divided into the following three groups: clause normalisation rules, resolution rules and extensional higher-order pre-unification rules. The set of clause normalisation rules are displayed in figure 1 and the resolution and unification rules of $\mathcal{ER}$ in figure 2. We want to point out that we assume commutativity of $\vee$ and $=$ and associativity of $\vee$. Furthermore we assume that after each application of a rule the generated clause is transformed into head-normal form.

[6] In the following we will use $\varphi * \mathbf{A}$ as an abbreviation for $\varphi \cup \{A\}$.

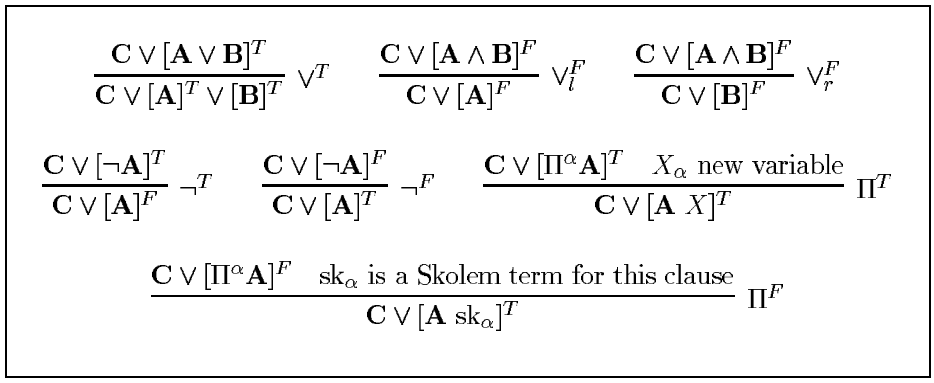


Figure 1: Clause normalisation rules in calculus $\mathcal{CNF}$

We want to remark that in the formal proofs we do not assume idem potence of $\vee$ and do not factorise identical literals in the clause normalisation process as this eases our argumentations. In practice one is certainly interested to optimise clause normalisation as far as possible.

Before we discuss the rules of $\mathcal{ER}$ in detail, we briefly sketch the idea of traditional first-order resolution [Rob65]: First-order resolution can be seen as two layered approach where the overall search for a refutation based on the resolution rule *resolve* and the factorisation rule *factorise*) is performed on the upper layer. The upper layer passes subproblems to the lower layer, such as the initial clause normalisation process or the intermediate unification problems. Very important is that all the side computations performed on the lower layer are decidable.

In our higher-order setting clause normalisation remains uncritical and the set of clauses $\mathcal{CNF}(\Phi)$ for a given set of higher-order formulas Φ can easily be computed with the clause normalisation rules[7]. Thus in calculus $\mathcal{ER}$ clause normalisation is still employed as a side computation (evoked by rule *Cnf*) whenever it seems to be appropriate.

In contrast to clause normalisation higher-order unification is undecidable [GS89] and thus higher-order unification can no longer be employed as a filtering side computation like in the traditional first-order setting. Huet solved this problem in [Hue72, Hue73a] by delaying unification in his original constraint resolution approach instead of employing it as a filter. As a resolution approach that generally delays the unification filter until an empty clause is derived can certainly not form the basis of an efficient higher-order theorem prover, Kohlhase allows within in his sorted variant of Huet's resolution calculus $\mathcal{HORES}$ (see [Koh94]) for eager unification (note that in practice many unification constraints have none or only finitely many solutions and that this information can positively influence the search space).

The unsorted variant of $\mathcal{HORES}$ also provides the basis for the calculi $\mathcal{ER}$, $\mathcal{EP}$ and $\mathcal{ERUE}$ discussed in this paper. But unfortunately $\mathcal{HORES}$ as presented in [Koh94] is neither sound (e.g. one can proof that any function has a fix-point in $\mathcal{HORES}$) nor complete for functional Σ-models[8] (or Henkin models) This is the main motivation for the modifications of $\mathcal{ER}$ over $\mathcal{HORES}$. For instance, as the author did not find a way to correct the extra logical treatment of Skolemisation in calculus $\mathcal{HORES}$ (which annotates special variable conditions to each clause) we will use traditional Skolemisation again; more precisely we will use Miller's sound approach for higher-order Skolemisation [Mil83]. Another, more notational modification belongs to the encoding of unification constraints which we uniformly present as negated equations in all new calculi. And the most important modification is that we add new extensionality rules to the calculus $\mathcal{HORES}$

[7] The clause normalisation algorithm that transforms a set Φ of formulas or a pre-clause $\mathcal{C}$ into respective clause sets $\mathcal{CNF}(\Phi)$ (or $\mathcal{CNF}(\mathcal{C})$) proceeds as follows: Initially all formulas $\mathbf{A} \in \Phi$ are replaced by pre-clause $[\mathbf{A}_{\downarrow_h}]^T$ and then the clause normalisation rules are exhaustively applied to Φ.

[8] This notion of semantics is weaker than Henkin-semantics and for instance the boolean extensionality property is not valid in all functional Σ-models. On the other hand the functional extensionality principle still holds in functional Σ-models but $\mathcal{HORES}$ is not able to show this.

Clause Normalisation: (defined for arbitrary clauses)

$$\frac{\mathcal{D} \quad \mathcal{C} \in \mathcal{CNF}(\mathcal{D})}{\mathcal{C}}\ Cnf$$

Resolution: (defined on proper clauses only)

$$\frac{[\mathbf{A}]^\alpha \vee \mathbf{C} \quad [\mathbf{B}]^\beta \vee \mathbf{D} \quad \alpha \neq \beta}{\mathbf{C} \vee \mathbf{D} \vee [\mathbf{A} = \mathbf{B}]^F}\ Res \qquad \frac{[\mathbf{A}]^\alpha \vee [\mathbf{B}]^\alpha \vee \mathbf{C} \quad \alpha \in \{T, F\}}{[\mathbf{N}]^\alpha \vee \mathbf{C} \vee [\mathbf{A} = \mathbf{B}]^F}\ Fac$$

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

Extensional (Pre-)Unification: (defined for arbitrary clauses)

$$\frac{\mathbf{C} \vee [\mathbf{M}_{\alpha\to\beta} = \mathbf{N}_{\alpha\to\beta}]^F \quad s_\alpha \text{ Skolem term for this clause}}{\mathbf{C} \vee [\mathbf{M}\ s = \mathbf{N}\ s]^F}\ Func$$

$$\frac{\mathbf{C} \vee [\mathbf{A}_{\alpha\to\beta}\ \mathbf{C}_\alpha = \mathbf{B}_{\alpha\to\beta}\ \mathbf{D}_\alpha]^F}{\mathbf{C} \vee [\mathbf{A} = \mathbf{B}]^F \vee [\mathbf{C} = \mathbf{D}]^F}\ Dec$$

$$\frac{\mathbf{C} \vee [\mathbf{A} = \mathbf{A}]^F}{\mathbf{C}}\ Triv \qquad \frac{\mathbf{C} \vee [X = \mathbf{A}]^F \quad X \notin \mathbf{Free}(\mathbf{A})}{\mathbf{C}_{\{\mathbf{A}/X\}}}\ Subst$$

$$\frac{\mathbf{C} \vee [F_\gamma\ \overline{\mathbf{U}^n} = h\ \overline{\mathbf{V}^m}]^F \quad \mathbf{G} \in \mathcal{GB}_\gamma^h}{\mathbf{C} \vee [F = \mathbf{G}]^F \vee [F\ \overline{\mathbf{U}^n} = h\ \overline{\mathbf{V}^m}]^F}\ FlexRigid$$

$$\frac{\mathbf{C} \vee [\mathbf{M}_o = \mathbf{N}_o]^F}{\mathbf{C} \vee [\mathbf{M}_o \Leftrightarrow \mathbf{N}_o]^F}\ Equiv \qquad \frac{\mathbf{C} \vee [\mathbf{M}_\alpha = \mathbf{N}_\alpha]^F}{\mathbf{C} \vee [\forall P_{\alpha\to o}.\ P\ M \Rightarrow P\ N]^F}\ Leib$$

Figure 2: Extensional Higher-Order Resolution calculus $\mathcal{ER}$

in order to reach Henkin completeness. The rules of $\mathcal{HORES}$ that are directly reflected in $\mathcal{ER}$ are the clause normalisation rules presented in figure 1, the resolution and the pre-unification rules — except for the extensionality rules *Equiv* and *Leib* — as stated in figure 2.

We now discuss the calculus $\mathcal{ER}$ presented in figure 2 in detail and start with those rules that are imported from in Kohlhase's extensionally incomplete calculus $\mathcal{HORES}$[9] The higher-order resolution rule (*Res*) and factorisation rule (*Fac*) employed in $\mathcal{ER}$ (and $\mathcal{HORES}$) differ from their first-order counterparts. Instead of using unification as a filter they add – as suggested by Huet – respective unification constraints to the generated clauses. As already mentioned, our calculus allows for eager pre-unification. This is realised by rule *Subst* which propagates (early) solved unification constraints back to the other literals. After applying this rule, clause normalisation may become necessary in order to obtain proper clauses again. This is due to the fact that instantiating predicate variables at head positions of some literals (flexible heads) may lead to pre-clauses instead of proper ones. The clause normalisation process is evoked by the application of rule *Cnf* which hides and groups each exhaustive $\mathcal{CNF}$-derivation of a proper clause $\mathcal{C}$ from a pre-clause $\mathcal{C}'$ with the $\mathcal{CNF}$-rules in only one rule application of *Cnf* in the calculus $\mathcal{ER}$.

It is well known for higher-order resolution (see [And71]) that primitive substitution rule *Prim* is needed as unification is to weak within a resolution approach to compute all necessary instantiations for flexible literal heads. This rule allows to instantiate flexible literal heads by a general binding that imitates a logical constant given in our signature. The importance of this rule can be illustrated by the example $\exists X_o \exists Y_o.\ X \vee Y$ which is certainly a theorem with respect to Henkin semantics. By negation and clause normalisation we obtain the two unit clauses $[X]^F$ and $[Y]^F$. Both clauses consists of exactly one negated literal with a flexible head. Neither resolution nor factorisation is applicable and thus without the primitive substitution rule we cannot find a refutation. The application of *Prim* with the general binding $\{X \leftarrow \neg X'\}$ on clause $[X]^F$ results after clause normalisation with rule *Cnf* in $[X']^T$. Thus, with primitive substitution rule *Prim* we add important but probably missing logical structure to our clauses, such that a refutation with the calculus rules becomes possible.

Despite the undecidability of higher unification the overall idea of calculus $\mathcal{ER}$ (and $\mathcal{HORES}$) is to apply unification rules as early as possible in order to filter out all those clauses with non-unifiable constraints or to propagate obvious variable instantiations from the unification constraints to the other literals of the clause.

Unfortunately $\mathcal{HORES}$ lacks completeness with respect to Henkin semantics. The problem is that despite the primitive substitution rule which in some sense supports the embedded higher-order unification algorithm the unification rules are still to weak to handle the extensionality principles sufficiently. More precisely, higher-order unification as employed in $\mathcal{HORES}$ or in Huet's original approach is a pure syntactically oriented algorithm for equalising terms. But what we really need is unification with respect to the theory defined by the extensionality principles. For example, in higher-order theorem proving we are also interested to unify terms like $\mathbf{A}_o \wedge \mathbf{B}_o$ and $\mathbf{B}_o \wedge \mathbf{A}_o$ or even $\lambda X_\alpha.\ \mathbf{A}_{\alpha\to o}\ X \wedge \mathbf{B}_{\alpha\to o}\ X$ and $\lambda X_\alpha.\ \mathbf{B}_{\alpha\to o}\ X \wedge \mathbf{A}_{\alpha\to o}\ X$.

In order to reach Henkin completeness calculus $\mathcal{HORES}$ as well as Huet's or Andrews' higher-order resolution approaches require the extensionality axioms to be added to the search space. But this is certainly awkward to handle in practice as these additional axioms on the one hand introduce many flexible heads in the search space and on the other hand negatively influence the few remaining goal oriented aspects of higher-order resolution.

Therefore calculus $\mathcal{ER}$ extends calculus $\mathcal{HORES}$ and provides three new extensionality rules. The extensionality rule *Leib* (see figure 2) simply instantiates the equality symbol within unification constraints by its Leibniz definition and rule *Equiv* which is directly motivated by the proof attempt of **E1** discussed in [BK98a] reflects the extensionality property for truth values but in a negative way: if two formulas are not equal then they are also not equivalent. Rule *Func* analogously reflects functional extensionality: if two functions are not equal then there exists an argument s_α on which

[9] [Koh95] presents a tableaux calculus analogous to $\mathcal{HORES}$ that aims to reach Henkin completeness without extensionality by adding rule *Equiv*. But unfortunately this calculus, which also influenced the work presented here, still lacks Henkin completeness as it not realises a suitable interaction of the functional and boolean extensionality principles.

these functions differ. To ensure soundness, s_α has to be a new Skolem term which contains all the free variables occurring in the given clause. The reader may be astonished why rule *Func* is also presented as usual unification rule here. The simple reason is that the pre-unification rules α and η as presented in $\mathcal{HORES}$ (and which are still explicitly mentioned in [BK98a]) already partly realize the negative aspect of the functional extensionality principle:

$$\frac{C \vee [(\lambda X_\alpha.\ \mathbf{A}) = (\lambda Y_\alpha.\ \mathbf{B})]^F \qquad s_\alpha \text{ Skolem term for this clause}}{C \vee [\{s/X\}\mathbf{A} = \{s/Y\}\mathbf{B}]^F}\ \alpha$$

$$\frac{C \vee [(\lambda X_\alpha.\ \mathbf{A}) = \mathbf{B}]^F \qquad s_\alpha \text{ Skolem term for this clause}}{C \vee [\{s/X\}\mathbf{A} = (\mathbf{B}s)]^F}\ \eta$$

Note that the purely type information based rule *Func* extends and generalises these two rules and thus rule Func has the following two meanings in our calculus: On the one hand it is a rule that functions like the traditional α- and η-rule as for instance used in [BK98a] and [Koh94]. On the other hand *Func* realizes the functional extensionality principle (especially when the unification constraint in focus is not of one of the forms required in the α- or η-rule), i.e. both hand sides are non-abstraction terms.

In cooperation the new extensionality rules connect the unification part of our calculus with the resolution part by allowing for recursive calls of the whole calculus from within the unification process. This turns our calculus into an test algorithm for extensional higher-order E-pre-unification.

We want to point out that none of the three new extensionality rules introduces any flexible literal and even better, they introduce no new free variable at all; even if they heavily increase the search space for refutations, they behave much better – as experiments show with the LEO theorem prover [BK98b, Ben97] – than the extensionality axioms, which introduce lots of flexible literals in the refutation process.

One important aspect that is illustrated by the examples in this paper as well as the examples discussed in [BK98a] is that eager pre-unification is essential in our extensional resolution approach and many proofs cannot be found when delaying the unification process till the end. Aside from the new extensionality rules this the most important difference to Huet's original approach.

We have lifted the unification constraints to clause level by coding them into negated equation literals. Hence the question arises whether or not resolution and factorisation rules are allowed to be applied on these unification constraints. In order to obtain a Henkin complete calculus this is not necessary – as the completeness proofs in [BK98a, BK97a] and the alternative one in this paper shows. Consequently the unification constraints do not necessarily have to be coded as negative equation literals, any other form will work as well. But the encoding of unification constraints as negated equation literals is essential for the extensional higher-order RUE-resolution calculus presented in sections 6.

We have already mentioned that the presentation of the extensional higher-order resolution approach $\mathcal{ER}$ slightly differs from the presentation in [BK98a] and we briefly sum up the particular modifications:

- The unification rules α and η employed in [BK98a] are avoided as they are subsumed by rule *Func*.
- Instead of employing clause normalisation in the definitions of rule *Subst* and the extensionality rules *Equiv* and *Leib* we add the extra clause normalisation rule *Cnf* to the calculus.
- We slightly modify the decomposition rule *Dec*. This modification is illustrated in detail by the example given in subsection 7.1.

For another discussion of the calculus $\mathcal{ER}$ and some simple but interesting proof examples we refer to [BK98a, Ben97]. The calculus is implemented in the LEO-theorem prover [BK98b, Ben97] which is integrated to the mathematical assistant OMEGA [BCF+97].

In this paper we will first present an alternative proof for the Henkin completeness of calculus $\mathcal{ER}$ to the one presented in [BK98a, BK97a]. The motivation for this new proof is threefold:

(i) We have slightly modified the calculus $\mathcal{ER}$ in this paper. (ii) The completeness proofs of the new, further extended calculi $\mathcal{EP}$ (extensional higher-order paramodulation) and $\mathcal{ERUE}$ (extensional higher-order RUE-Resolution) are carried out analogously, such that many lemmata can be directly reused or with minor modifications. (iii) The lifting lemma in the completeness proofs in [BK98a, BK97a] builds up on an argumentation also employed in [Koh94] which uses a rather complicate notion of clause isomorphisms susceptible to errors. In this paper we present a lifting argument that omits this rather complicate and unintuitive notion. This is possible as we analyse a generalised resolution calculus $\mathcal{ER}_{fc}$ instead of $\mathcal{ER}$. This enriched calculus additionally employs the instantiation guessing *FlexFlex*-rule (see figure 3) and applies the single clause normalisation rules instead of grouping them into exhaustive clause normalisation chains with rule *Cnf*. Consequently, in subsection 3.5 we will discuss the theorem equivalence between $\mathcal{ER}_{fc}$ and $\mathcal{ER}$.

An important convention for this and the following sections concerns α-equality of clauses and the arity of Skolem terms:

Remark 3.1 ***(Equality of clauses).*** In resolution based theorem proving one usually assumes all clauses to be variable disjunct. In practice this is achieved by automatically renaming the variables within each freshly generated clause. In this paper we implicitly use this convention too.

Another implicit convention concerns the Skolem terms. We briefly illustrate this aspect by an example. Assume that the following pre-clause is given:

$$\mathcal{C}_1 : [\forall X_\iota .\ p_{\iota\to o}\ X\ Y_\iota]^T \vee [\forall Z_\iota .\ q\ Z]^F$$

Clause normalisation either leads to

$$\mathcal{C}_2 : [p_{\iota\to o}\ X\ Y_\iota]^T \vee [q\ (s\ Y)]^F \quad \text{or to} \quad \mathcal{C}_3 : [p_{\iota\to o}\ X\ Y_\iota]^T \vee [q\ (s\ X\ Y)]^F$$

where $(s_{\iota\to\iota}\ Y)$ and $(s_{\iota\to\iota\to\iota}\ X\ Y)$ are Skolem terms. The first clause $\mathcal{C}_2$ is the result of applying rule Π^T first and Π^F to the result, whereas the second clause $\mathcal{C}_3$ is the result of applying first Π^F and then Π^T. Both results differ with respect to the arity of the new Skolem terms. It is well known for refutation approaches that each refutation using only one of this clauses can be analogously carried out with the other one. For a discussion of this Skolemisation aspect in the context of of sequent calculi we refer to [AMS98]. In the following we will therefore ignore the different arities of Skolem terms caused by switching the order of single applications of clause normalisation rules (switching the order of clause normalisation rules will be employed in some of proofs within this paper).

For those readers which are still not familiar with the main ideas of our extensional higher-order resolution approach we refer to the illustrating example in subsection 7.1 or the examples discussed in [BK98a]. A good intuition of the practical usage of the extensional higher-order resolution approach will support a better understanding of the following formal proofs.

3.2 Basic Definitions

We already pointed out that instead of an direct discussion of Henkin completeness for $\mathcal{ER}$ we first analyse the slightly enriched calculus $\mathcal{ER}_{fc}$. Aside from the ungrouping of exhaustive clause normalisation derivations this calculus provides the well known *FlexFlex* rule displayed in figure 3. In case a clause contains a *flex-flex*-unification constraint this rule allows to guess a instantiation for one of the flexible heads such that the unification process can proceed with its eager unification attempts. It was already pointed out by Huet [Hue72] that in practice we are interested to avoid this possibly infinitely branching rule (there may be infinitely many constants in the signature) and it turned out that within a refutation approach one can in fact do without the *FlexFlex*-rule and delay the operations on *flex-flex*-constraints until one of the head variables gets bound. On the other hand employing this additional rule within the lifting lemma eases the proofs as it turns our eager pre-unification approach into an eager unification approach. This makes it possible to get rid of the clause isomorphisms that are needed in the analogous proofs in [Koh94]. The motivation for the ungrouping of exhaustive clause normalisation derivation by rejecting rule *Cnf* and Lifting the

$$\frac{\mathbf{C} \vee [F_{\overline{\gamma^n}\to\alpha}\ \overline{\mathbf{U}^n} = H_{\overline{\delta^m}\to\alpha}\ \overline{\mathbf{V}^m}]^F \quad \mathbf{G} \in \mathcal{GB}^h_{\overline{\gamma^n}\to\alpha} \text{ for a } h_\tau \in \mathcal{C}_\tau}{\mathbf{C} \vee [F\ \overline{\mathbf{U}^n} = h\ \overline{\mathbf{V}^m}]^F \vee [F = \mathbf{G}]^F}\ FlexFlex$$

Figure 3: The *FlexFlex* rule

clause normalisation rules to calculus level is the same: We want to ease the proofs in this section and especially the analogous but slightly more complicate ones for the extensional higher-order paramodulation calculus $\mathcal{EP}$ and extensional higher-order RUE-resolution calculus $\mathcal{ERUE}$.

Unfortunately a formal proof for the admissibility of rule *FlexFlex* has not been carried out yet but evidence is given by the direct proof of Henkin completeness in [BK98a] (which admittedly lacks a bit of transparency and clarity within the lifting argument) and the case studies with the LEO prover [BK98b] which implements extensional higher-order resolution approach. All examined examples so far do not require the application of rule *FlexFlex*, although many of them require eager (pre-)unification.

Definition 3.2 (Clause Normalisation). The calculus $\mathcal{CNF}$ consists of the clause normalisation rules displayed in figure 1. We assume that the result of each rule application is transformed into head-normal form.[10] By exhaustively applying these rules to a pre-clause $[\mathbf{A}]^\alpha$ ($\alpha \in \{T, F\}$ one can derive the set $\mathcal{CNF}([\mathbf{A}]^\alpha)$ of proper clauses derivable from $[\mathbf{A}]^\alpha$.

Lemma 3.3 (Soundness of $\mathcal{CNF}$). *The rules $\mathcal{CNF}\backslash\{\Pi^F\}$ preserve validity and the rule Π^F preserves satisfiability with respect to standard models.*

Proof: The proofs for the rules in $\mathcal{CNF}\backslash\{\Pi^F\}$ are analogous to the first-order case and Skolemisation has been corrected for higher-order logic by Miller in [Mil83, Mil91, Mil92]. □

Definition 3.4 (Unification). We define the following two calculi for higher-order unification and higher-order pre-unification:

- $\mathcal{UNI}$ The calculus $\mathcal{UNI}$ consists of the pre-unification rules *Triv, Func, Dec, FlexRigid* and *Subst*. We assume that the result of each rules application is transformed into head-normal normal form.[11] These rules are a rather close variant of the higher-order pre-unification calculus discussed in [Koh94] and [GS89]. As discussed before, rule *Func* subsumes the rules α and η used in [BK98a] and [Koh94].
- $\mathcal{UNI}_f$ The calculus $\mathcal{UNI}_f$ is defined as $\mathcal{UNI} \cup \{FlexFlex\}$.

Theorem 3.5 (Higher-Order Unification and Pre-Unification).

1. *$\mathcal{UNI}$ is a sound and complete calculus for higher-order pre-unification.*
2. *$\mathcal{UNI}_f$ is a sound and complete calculus for higher-order pre-unification.*

Proof: We will not present a formal proof here and refer to [Koh94] instead as the set of rules discussed there is very similar to the one discussed here. The only difference is that [Koh94] additionally examines sorts and uses a extra-logical form of Skolemisation. Note that the set of rules presented only slightly adapt the rules discussed in [GS89]. □

The first complete set for higher-order unification was defined in [Pie73] and undecidability of higher-order unification was first discussed in [Hue73b]. Huet then introduced higher-order pre-unification in [Hue75]. For a modern presentation of higher-order unification and pre-unification we refer to [GS89]. Sorted higher-order unification and pre-unification is discussed in [Koh94].

[10] Remember the special definition of head-normal form for unification constraints.

[11] Note that this is the main difference to the pre-unification rules presented in [GS89] which presupposes that all results are reduced to $\beta\eta$-normal form.

As the calculus $\mathcal{UNI}_f \backslash \{Equiv, Leib\}$ realises higher-order unification and as rule *Subst* allows to propagate solutions back to the non-unification constraints of a clause we get the following corollary.

Corollary 3.6 (Higher-Order Unification). *Let $\mathcal{C} \vee E$ be a clause with unification constraints E. Then for each unifier σ of E we have that $\mathcal{C} \vee E \vdash_{\mathcal{UNI}_f} \mathcal{C}\sigma$.*

Definition 3.7 (Extensional Higher-Order Resolution).

- $\mathcal{ER}$ The calculus $\mathcal{ER}$ consists of the following inference rules displayed in figure 2, i.e. $\mathcal{ER} := \{Cnf, Res, Fac, Prim\} \cup \mathcal{UNI}$
- $\mathcal{ER}_f$ The extension $\mathcal{ER}_f$ of calculus $\mathcal{ER}$ that employs full higher-order unification instead of higher-order pre-unification is defined as $\mathcal{ER}_f := \mathcal{ER} \cup \{FlexFlex\}$.
- $\mathcal{ER}_{fc}$ The calculus $\mathcal{ER}_{fc}$ that employs stepwise instead of exhaustive clause normalisation is defined by $\mathcal{ER}_{fc} := (\mathcal{ER}_f \backslash \{Cnf\}) \cup \mathcal{CNF}$.

For all calculi we assume that the result of each rules application is transformed into head-normal normal form. A set of formulas Φ is refutable in calculus $R \in \{\mathcal{ER}, \mathcal{ER}_f, \mathcal{ER}_{fc}\}$ iff there is a derivation $\Delta : \Phi_{cl} \vdash_R \square$ where $\Phi_{cl} := \{[\mathbf{F}_{\downarrow_h}]^T | \mathbf{F} \in \Phi\}$ is the set obtained from Φ by simple pre-clausification. We again point out that unification constraints are treated as special literals which are only accessible to the unification rules.

*Remark 3.8 (**General Higher-Order E-Unification**).* The extensional higher-order resolution calculus $\mathcal{ER}$ ($\mathcal{ER}_f$ or $\mathcal{ER}_{fc}$) can also be interpreted as a test calculus for general higher-order E-pre-unifiability (E-unifiability): Assume an arbitrary set of equations $\mathbf{E}_1 \ldots \mathbf{E}_n$ describing a theory E and a E-unification problem $\mathbf{T}_1 = \mathbf{T}_2$ is given. If we pass clauses $[\mathbf{E}_1]^T \ldots [\mathbf{E}_n]^T$ and $[\mathbf{T}_1 = \mathbf{T}_2]^F$ as an input problem to our calculus, then the calculus $\mathcal{ER}$ tests if the unification constraint $[\mathbf{T}_1 = \mathbf{T}_2]^F$ is solvable with respect to theory E enriched by the extensionality properties. The overall answer substitution computed by the refutation is obviously also an answer substitution to our E-unification problem.

Theorem 3.9 (Soundness of Extensional Higher-Order Resolution). *The calculi $\mathcal{ER}$, $\mathcal{ER}_f$ and $\mathcal{ER}_{fc}$ are sound.*

Proof: We already know by lemma 3.3 that the rules in $\mathcal{CNF}$ either preserve validity or satisfiability with respect to standard models and thus the latter also holds for rule *Cnf*. All unification rules apart from *Func* as well as the rules *Res*, *Fac* can easily be shown to preserve validity with respect to standard models. And it is easy to check that the rules *Func* (which simply employs Skolemisation) and *Prim* (which instantiates a free variable) preserve satisfiability. Now the assertion follows from the well known result that preservation of satisfiability ensures soundness within a refutation approach. □

Lemma 3.10 (Proper Derivations). *For each non proper clause $\mathcal{C}$, proper clause $\mathcal{C}'$ and clause set Φ such that $\Phi * \mathcal{C} \vdash_{\mathcal{ER}_{fc}} \mathcal{C}'$ we have that $\Phi \cup \mathcal{CNF}(\mathcal{C}) \vdash_{\mathcal{ER}_{fc}} \mathcal{C}'$. This statement analogously holds for calculus $\mathcal{ER}$.*

Proof: The proof is by induction on the length of derivation $\Phi * \mathcal{C} \vdash_{\mathcal{ER}_{fc}} \mathcal{C}'$. In the base case ($n \equiv 0$) we know that that $\mathcal{C}' \in \Phi$ as $\mathcal{C}'$ must be different from $\mathcal{C}$. Thus the assertion follows trivially. In the induction step ($n > 0$) we consider the first step in derivation $\Phi * \mathcal{C} \vdash^r \Phi * \mathcal{C} * \mathcal{D} \vdash_{\mathcal{ER}_{fc}} \mathcal{C}'$ If $\mathcal{C}$ is not a premise clause for the application of rule r then the assertion follows immediately by induction hypotheses. Thus, let us assume that $\mathcal{C}$ is a premise clause for the application of rule r. By induction hypotheses applied to clause $\mathcal{D}$ we get that $(\Phi * \mathcal{C}) \cup \mathcal{CNF}(\mathcal{D}) \vdash_{\mathcal{ER}_{fc}} \mathcal{C}'$. As the resolution rules $Res, Prim, Fac$ are defined on proper clauses only we know that the only rules that are possible for r are the unification rules in $\mathcal{UNI}_f$ and the rule Cnf. It is easy to check that in all those cases we have that $\mathcal{CNF}(\mathcal{C}) \vdash_{\mathcal{ER}_{fc}} \mathcal{CNF}(\mathcal{D})$. And thus, we finally get that $\Phi \cup \mathcal{CNF}(\mathcal{C}) \vdash_{\mathcal{ER}_{fc}} \Phi \cup \mathcal{CNF}(\mathcal{C}) \cup \mathcal{CNF}(\mathcal{D}) \vdash_{\mathcal{ER}_{fc}} \mathcal{C}'$. □

3.3 Lifting Properties

This subsection examines some lifting properties of calculus $\mathcal{ER}_{fc}$. First we will prove a lifting argument for all clause normalisation rules in $\mathcal{CNF}$ before we then present the lifting lemma for $\mathcal{ER}_{fc}$. The lifting lemma for calculus $\mathcal{ER}_{fc}$ turns out to be rather trivial. The reason is that $\mathcal{ER}_{fc}$ provides rule *FlexFlex* and that the clause normalisation rules are not grouped into one single rule *Cnf* but treated as single inference rules for their own. The main idea in the lifting lemma for calculus $\mathcal{ER}_{fc}$ is to show that for each derivation step performed on an instantiated set of clauses there exists an analogous step (or multiple step derivation) on the abstract level as well. And in blocking situations — this is when the abstract, uninstantiated level does not contain enough structural information as some free variables occur at head positions of some literals or at head position in one of the two terms in an unification constraint — then either rule *Prim* or rule *FlexFlex* can be used to unblock the situation by restricting the free head variables in an appropriate but still most general way.

Lemma 3.11 (Lifting of Clause Normalisation). *Let the $\mathcal{D}_1, \mathcal{D}_2$ be clauses and σ a be substitutions. For each derivation $\Delta_1 : (\mathcal{D}_1)_\sigma \vdash^1_{\mathcal{CNF}} \mathcal{D}_2$ exists a clause $\mathcal{D}_3$, a substitution δ and a derivation $\Delta_2 : \mathcal{D}_1 \vdash_{\mathcal{ER}_{fc}} \mathcal{D}_3$ such that $(\mathcal{D}_3)_{\sigma\circ\delta} \equiv \mathcal{D}_2$.*

Proof: The proof is by case distinction on the rules in $\mathcal{CNF}$. As all cases are analogous we only consider the $\vee^F_l$ rule here. In this case we have that $(\mathcal{D}_1)_\sigma$ and $\mathcal{D}_2$ have the form $(L_1)_\sigma \vee \ldots \vee (L_n)_\sigma \vee [\mathbf{A}_\sigma \vee \mathbf{B}_\sigma]^F$ and $(L_1)_\sigma \vee \ldots \vee (L_n)_\sigma \vee [\mathbf{A}_\sigma]^F$ for some literals $(L_i)_\sigma$ and terms $\mathbf{A}_\sigma, \mathbf{B}_\sigma, \mathbf{C}_\sigma$. We now consider the possible structure of the focused literal in the uninstantiated clause $\mathcal{D}_1 : L_1 \vee \ldots \vee L_n \vee [\mathbf{C}]^F$. In case $[\mathbf{C}]^F \equiv [\mathbf{A} \vee \mathbf{B}]^F$ the we can obviously apply $\vee^F_l$ to this clause leading to $\mathcal{D}_3 : L_1 \vee \ldots \vee L_n \vee [\mathbf{A}]^F$. Then the assertion follows trivially. In the other possible case we have $[\mathbf{C}]^F \equiv [H\ \overline{\mathbf{U}^m}]^F$ such that $(H\ \overline{\mathbf{U}^m})_\sigma \equiv \mathbf{A}_\sigma \vee \mathbf{B}_\sigma$, $H_\sigma \equiv \lambda\overline{X^m}.\, \mathbf{L} \vee \mathbf{R}$, $(\mathbf{L}[\overline{\mathbf{U}^m}/\overline{X^m}])_\sigma \equiv \mathbf{A}_\sigma$ and $(\mathbf{R}[\overline{\mathbf{U}^m}/\overline{X^m}])_\sigma \equiv \mathbf{B}_\sigma$. We then apply rule *Prim* to clause $\mathcal{D}_1$ with general binding $\lambda\overline{X^m}.\, (H_1\ \overline{X^m}) \vee (H_2\ \overline{X^m})$ for predicate variable H (H^1, H^2 are new predicate variables of appropriate type) and obtain clause $\mathcal{D}_4 : L_1 \vee \ldots \vee L_n \vee [\mathbf{C}]^F \vee [H = \lambda\overline{X^m}.\, (H_1\ \overline{X^m}) \vee (H_2\ \overline{X^m})]^F$. With rule *Subst* we get clause $\mathcal{D}_5 : (L_1)_\tau \vee \ldots \vee (L_n)_\tau \vee [(H^1\ \overline{\mathbf{U}^m}) \vee (H^2\ \overline{\mathbf{U}^m})]^F$ where $\tau := [\lambda\overline{X^m}.\, (H_1\ \overline{X^m}) \vee (H_2\ \overline{X^m})/H]$. Now rule $\vee^F$ is applicable which leads to clause $\mathcal{D}_3 : (L_1)_\tau \vee \ldots \vee (L_n)_\tau \vee [(H^1\ \overline{\mathbf{U}^m})]^F$. It is easy to verify that τ is more general than σ and that hence there must be a substitution γ that appropriately instantiates the new predicate variables H^1 and H^2 and that coincides with σ on all other variables such that we have $(L_i)_{\gamma\circ\tau} \equiv (L_i)_\sigma$, $(H^1\ \overline{\mathbf{U}^m})_{\gamma\circ\tau} \equiv \mathbf{A}_\sigma$. This finally proves the assertion as $(\mathcal{D}_3)_{\gamma\circ\tau} \equiv \mathcal{D}_2$. □

We are now ready to prove the lifting lemma for calculus $\mathcal{ER}_{fc}$. The main statement (see statement 3.12(2)) is that for any substitution σ and clause set Φ holds that Φ is refutable in calculus $\mathcal{ER}_{fc}$ provided that Φ_σ is.

Lemma 3.12 (Lifting Lemma for $\mathcal{ER}_{fc}$). *Let Φ be a set of clauses, $\mathcal{D}_1$ be a clause and σ a substitution. We have that:*

1. *For each derivation $\Delta_1 : \Phi_\sigma \vdash^1_{\mathcal{ER}_{fc}} \mathcal{D}_1$ there exists a substitution δ, a clause $\mathcal{D}_2$ and a derivation $\Delta_2 : \Phi \vdash_{\mathcal{ER}_{fc}} \mathcal{D}_2$ such that $(\mathcal{D}_2)_\delta \equiv \mathcal{D}_1$.*

2. *For each derivation $\Delta_1 : \Phi_\sigma \vdash_{\mathcal{ER}_{fc}} \mathcal{D}_1$ there exists a substitution δ, a clause $\mathcal{D}_2$ and a derivation $\Delta_2 : \Phi \vdash_{\mathcal{ER}_{fc}} \mathcal{D}_2$ such that $(\mathcal{D}_2)_\delta \equiv \mathcal{D}_1$.*

Proof:

(1) The proof is by case distinction on all rules in $\mathcal{ER}_{fc}$ and in all cases we will motivate that there is a derivation Δ_2 as required.

Res Assume the first step in Δ_1 employs resolution rule *Res* to clauses $[\mathbf{A}_\sigma]^\alpha \vee C_\sigma$ and $[\mathbf{A}_\sigma]^\beta \vee D_\sigma$ with resolvent $\mathcal{D}_1 : C_\sigma \vee D_\sigma \vee [\mathbf{A}_\sigma = \mathbf{B}_\sigma]^F$. Then an analogous resolution step is possible between $[\mathbf{A}]^\alpha \vee C$ and $[\mathbf{A}]^\beta \vee D$ leading to resolvent $\mathcal{D}_2 : C \vee D \vee [\mathbf{A} = \mathbf{B}]^F$. We trivially have that $(\mathcal{D}_2)_\sigma \equiv C_1$.

Prim Assume the first step in Δ_1 employs rule *Prim* to a flexible literal in a clause $C_\sigma \vee [H\ \overline{(\mathbf{A}_\sigma)^n}]^\alpha$ leading to clause $\mathcal{D}_1 : C_\sigma \vee [H\overline{(\mathbf{A}_\sigma)^n}]^\alpha \vee [H = \mathbf{G}]^F$, where G is a general binding for variable H imitating a logical connective. Then an analogous proof step with an identical general binding is possible on the uninstantiated clause $C \vee [H\ \overline{\mathbf{A}^n}]^\alpha$ leading to $\mathcal{D}_2 : C \vee [H\ \overline{\mathbf{A}^n}]^\alpha \vee [H = \mathbf{G}]$ such that $(\mathcal{D}_2)_\sigma \equiv \mathcal{D}_1$.

Fac This case is analogous to *Res*.

Triv, *Dec*, *Func*, *FlexFlex*, *Leib*, *Equiv* These case are all analogous to *Prim*. We want to point out that $=$ is a special symbol not available in the signature and thus the abstract literal must also have head $=$. Therefore all these rules must be applicable on the abstract level as well.

FlexRigid Whereas we know that the abstract literal must have head $=$ the abstract literal is possibly a *FlexFlex*-constraint. In this case we employ the additional rule *FlexFlex* and *Subst* in order to introduce the corresponding rigid head of the instantiated literal. Thereby the *FlexRigid* step performed on the instantiated level becomes possible on the abstract level as well. It is easy to verify that the resulting clause on the abstract level is more general than the instantiated counterpart.

FlexFlex Like *Prim*, this rule is obviously applicable on the abstract level as well.

Subst In this case the ground literal has form $[X = \mathbf{A}_\sigma]^F$. It is easy to verify that the corresponding abstract literal must have the form $[Y = \mathbf{A}]^F$ for a variable Y. It can not be of form $[(\lambda Z.\ X)\mathbf{B} = \mathbf{A}]^F$ as the left hand side is not in head-normal form. Therefore this case is analogous to case *Prim*.

$r \in \mathcal{CNF}$ By clause normalisation lifting lemma 3.11.

(2) The proof is by induction on the length of derivation Δ_1. The base case is trivial and in the induction step we first employ statement (1) and then the induction hypothesis. □

3.4 Completeness

In this subsection we focus on the main proof of Henkin completeness for calculus $\mathcal{ER}_{fc}$. First we will present a lemma stating that reduction to head-normal form is sufficient in calculus $\mathcal{ER}_{fc}$ (note our special definition of head-normal form for unification constraints as described in section 2). A second lemma provides important refutational properties of our calculus. Finally in the prove of theorem 3.16 we show that the set of propositions that can not be refuted in calculus $\mathcal{ER}_{fc}$ defines an abstract consistency property for Henkin models 2.2. The latter entails Henkin completeness for $\mathcal{ER}_{fc}$ by theorem 2.1(1).

Lemma 3.13 (Head-Normal Form). *Let Φ be a set of clauses. If $\Delta : \Phi_{\downarrow\beta\eta} \vdash_{\mathcal{ER}_{fc}} \square$ then $\Delta' : \Phi_{\downarrow h} \vdash_{\mathcal{ER}_{fc}} \square$*

Proof: The proof is by induction on the length n of Δ and the base case ($n \equiv 0$) is trivial. In the step case ($n > 0$) we consider the first derivation step in $\Phi_{\downarrow\beta\eta} \vdash^r \Phi' \vdash \square$. In case $r \in \{Res, Fac\}$ the assertion follows immediately by induction hypotheses as both rules are applicable independently from the structure of the literals atoms. In case $r \in \{Prim\} \cup \mathcal{CNF}$ the heads of the atoms obviously play an important role. But note that for each atom we have that the head symbols in the $\beta\eta$-normal form and the head-normal form coincide such that each rule r is applicable in the head-normal form case as well. Therefore we again get the assertion by induction hypotheses. In case $r \in \mathcal{UNI}$ our special definition of head-normal form for unification constraints (which requires both hand sides of the constraint to be reduced to head-normal form) ensures the assertion with an analogous argumentation as in the previous case. □

Remark 3.14 ***(Head-Normal Form).*** We want to point out that it is not essential for our calculi whether we keep our clauses and literals in head-normal form or $\beta\eta$-normal form. The reason why we have chosen reduction to head-normal form is that this is less expensive in practice. If the reader does not trust in head-normal form here he can choose $\beta\eta$-normal form instead. This difference will hardly influence any of the discussions in this paper.

Lemma 3.15. *Let Φ be a set of clauses and $\mathbf{A}, \mathbf{B}$ be formulas. We have that:*

1. *If $\Phi * [\mathbf{A}]^T \vdash_{\mathcal{ER}_{fc}} \Box$ and $\Phi * [\mathbf{B}]^T \vdash_{\mathcal{ER}_{fc}} \Box$ then $\Phi * [\mathbf{A} \vee \mathbf{B}]^T \vdash_{\mathcal{ER}_{fc}} \Box$.*
2. *If $\Phi * [\mathbf{A}]^T * [\mathbf{B}]^F \vdash_{\mathcal{ER}_{fc}} \Box$ and $\Phi * [\mathbf{A}]^F * [\mathbf{B}]^T \vdash_{\mathcal{ER}_{fc}} \Box$ then $\Phi * [\mathbf{A} \Leftrightarrow \mathbf{B}]^F \vdash_{\mathcal{ER}_{fc}} \Box$.*

Proof: (1) As $\mathbf{A}$ and $\mathbf{B}$ are sentences we can assume without loss of generality that the pre-clauses $[\mathbf{A}]^T$ and $[\mathbf{B}]^T$ are variable disjunct. We can replay derivation $\Phi * [\mathbf{A}]^T \vdash_{\mathcal{ER}_{fc}} \Box$ in context $\Phi * [\mathbf{A} \vee \mathbf{B}]^T$ such that we get for each clause $\mathcal{C} \in \mathcal{CNF}([\mathbf{B}]^T)$ $\Phi * [\mathbf{A} \vee \mathbf{B}]^T \vdash_{\mathcal{CNF}} [\mathbf{A}]^T \vee \mathcal{C} \vdash_{\mathcal{ER}_{fc}} \mathcal{C} \vee E$ for a set of *FlexFlex* constraints E containing no variables that occur in $[\mathbf{B}]^F$ or $\mathcal{C}$. By lemma 3.6 we know that each unifier σ of E can be derived in $\mathcal{UNI}_f$ and thus we get for each unifier σ of clause $\mathcal{C} \vee E$ that $\mathcal{C} \vee E \vdash_{\mathcal{UNI}_f} \mathcal{C}_\sigma$. Now the assertion follows by lemma 3.10 as $\Phi * [\mathbf{B}]^T \vdash_{\mathcal{ER}_{fc}} \Box$ and as the domain of σ contains none of the free variables in $\mathcal{C}$ and thus $\mathcal{C}_\sigma \equiv \mathcal{C}$ for each clause $\mathcal{C} \in \mathcal{CNF}([\mathbf{B}]^T)$.

(2) This statement (which is also well known from first-order) can be proven by a tedious but straightforward computation which we only roughly sketch here. From the two assumptions we get by lemma 3.10 that (i) $\Phi * \mathcal{CNF}([\mathbf{A}]^T) * \mathcal{CNF}([\mathbf{B}]^F) \vdash_{\mathcal{ER}_{fc}} \Box$ and (ii) $\Phi * \mathcal{CNF}([\mathbf{A}]^F) * \mathcal{CNF}([\mathbf{B}]^T) \vdash_{\mathcal{ER}_{fc}} \Box$. The idea now is to apply exhaustive clause normalisation to the clause $[\mathbf{A} \Leftrightarrow \mathbf{B}]^F$ and then to show that (iii) $\Phi * \mathcal{CNF}([\mathbf{A} \Leftrightarrow \mathbf{B}]^F) \vdash_{\mathcal{ER}_{fc}} \Box$. Note that $\mathcal{CNF}([\mathbf{A} \Leftrightarrow \mathbf{B}]^F) := \{\mathcal{C} \vee \mathcal{D} | \mathcal{C} \in \mathcal{CNF}([\mathbf{A}]^\alpha)$ *and* $\mathcal{D} \in \mathcal{CNF}([\mathbf{B}]^\alpha)$ *for* $\alpha \in \{T, F\}\}$. Thus,. the task is to show that (iii) is a consequence of (i) and (ii) which is possible, for instance, by simultaneous induction on the structure of $\mathbf{A}$ and $\mathbf{B}$. □

Theorem 3.16 (Completeness of $\mathcal{ER}_{fc}$). *The calculus $\mathcal{ER}_{fc}$ is complete with respect to Henkin models.*

Proof: We adapt the analogous proof given in [BK98a].

Let Γ_Σ be the set of Σ-sentences which cannot be refuted by the calculus $\mathcal{ER}_{fc}$ ($\Gamma_\Sigma := \{\Phi \subseteq cwff_o(\Sigma) | \Phi_{cl} \not\vdash_{\mathcal{ER}_{fc}} \Box\}$), then we show that Γ_Σ is a saturated abstract consistency class for Henkin models 2.2 which entails Henkin completeness for $\mathcal{ER}_{fc}$ by theorem 2.1(1).

In particular have to verify that Γ_Σ ensures the abstract consistency properties $\nabla_c, \nabla_\neg, \nabla_\beta, \nabla_\vee, \nabla_\wedge, \nabla_\forall, \nabla_\exists, \nabla_\flat, \nabla_\natural$. Furthermore we have to show that Γ_Σ is saturated.

- ∇_c Suppose that $\mathbf{A}, \neg\mathbf{A} \in \Phi$. Since $\mathbf{A}$ is atomic we have $\Phi_{cl} * [\mathbf{A}]^T * [\neg\mathbf{A}]^T \vdash_{Cnf} \Phi_{cl} * [\mathbf{A}]^T * [\mathbf{A}]^F$ and hence we can derive $\Box$ with *Res* and *Triv*. This contradicts our assumption.

In all of the remaining cases, we show the contrapositive, e.g. in the next case we prove, that for all $\Phi \in \Gamma_\Sigma$, if $\Phi * \neg\neg\mathbf{A} * \mathbf{A} \notin \Gamma_\Sigma$, then $\Phi * \neg\neg\mathbf{A} \notin \Gamma_\Sigma$, which entails the assertion.

- $\nabla_\neg$ Let us assume that $\Phi_{cl} * [\neg\neg\mathbf{A}]^T * [\mathbf{A}]^T \vdash_{\mathcal{ER}_{fc}} \Box$. We immediately get the assertion since $[\neg\neg\mathbf{A}]^T \vdash_{\mathcal{CNF}} [\mathbf{A}]^T$.
- $\nabla_\mathfrak{f}$ If $\Phi_{cl} * [\mathbf{A}]^T * [\mathbf{A}_{\downarrow_{\beta\eta}}]^T \vdash_{\mathcal{ER}_{fc}} \Box$, then we get that $\Phi_{cl} * [\mathbf{A}]^T \vdash_{\mathcal{ER}_{fc}} \Box$ by lemma 3.13 (note that $\mathbf{A}$ is assumed to be in hnf).
- $\nabla_\vee$ If $\Phi_{cl} * [\mathbf{A} \vee \mathbf{B}]^T * [\mathbf{A}]^T \vdash_{\mathcal{ER}_{fc}} \Box$ and $\Phi_{cl} * [\mathbf{A} \vee \mathbf{B}] * [\mathbf{B}]^T \vdash_{\mathcal{ER}_{fc}} \Box$, then $\Phi_{cl} * [\mathbf{A} \vee \mathbf{B}]^T \vdash_{\mathcal{ER}_{fc}} \Box$ by lemma 3.15(1).
- $\nabla_\wedge$ Analogous to $\nabla_\neg$ as $[\neg(\mathbf{A} \vee \mathbf{B})]^T \vdash_{\mathcal{CNF}} [\neg\mathbf{A}]^T$ and $[\neg(\mathbf{A} \vee \mathbf{B})]^T \vdash_{\mathcal{CNF}} [\neg\mathbf{B}]^T$.
- $\nabla_\forall$ Let $\Phi_{cl} * [\Pi^\alpha\ \mathbf{F}]^T * [\mathbf{F}\ \mathbf{A}]^T \vdash_{\mathcal{ER}_{fc}} \Box$ for each closed formula $\mathbf{A}$. By lifting lemma 3.12 we get that $\Phi_{cl} * [\Pi^\alpha\ \mathbf{F}]^T * [\mathbf{F}\ \mathbf{X}]^T \vdash_{\mathcal{ER}_{fc}} \Box$ for a new variable X and thus obviously $\Phi_{cl} * [\Pi^\alpha\ \mathbf{F}]^T \vdash_{\mathcal{ER}_{fc}} \Box$.
- $\nabla_\exists$ Let us assume that $\Phi_{cl} * [\neg\Pi\ \mathbf{F}]^T * [\neg\mathbf{F}\ w]^T \vdash_{\mathcal{ER}_{fc}} \Box$. Note that $\neg\Pi\ \mathbf{F}$ is a closed formula and furthermore that w does not occur in $\Phi_{cl} * [\neg\Pi\ \mathbf{F}]^T$. We get the assertion immediately as $[\neg\Pi\ \mathbf{F}]^T \vdash_{\mathcal{CNF}} [\neg\mathbf{F}\ w]^T$.
- $\nabla_\natural$ We show that if $\Phi_{cl} * [\neg(\mathbf{A} \doteq^o \mathbf{B})]^T * [\neg\mathbf{A}]^T * [\mathbf{B}]^T \vdash_{\mathcal{ER}_{fc}} \Box$ and $\Phi_{cl} * [\neg(\mathbf{A} \doteq^o \mathbf{B})]^T * [\mathbf{A}]^T * [\neg\mathbf{B}]^T \vdash_{\mathcal{ER}_{fc}} \Box$, then $\Phi_{cl} * [\neg(\mathbf{A} \doteq \mathbf{B})]^T \vdash_{\mathcal{ER}_{fc}} \Box$. Note that $\Phi_{cl} * [\neg(\mathbf{A} \doteq \mathbf{B})]^T \equiv \Phi_{cl} * [\neg\Pi(\lambda P.\ \neg P\ \mathbf{A} \vee P\ \mathbf{B})]^T \vdash_{\mathcal{CNF}} \Phi_{cl} * [r\ \mathbf{A}]^T * [r\ \mathbf{B}]^F$ where $r_{o \to o}$ is a new Skolem constant. Now consider the following derivation

$$\dfrac{\dfrac{\dfrac{[r\ \mathbf{A}]^T \quad [r\ \mathbf{B}]^F}{[r\ \mathbf{A} = r\ \mathbf{B}]^F}\ Res}{[\mathbf{A} = \mathbf{B}]^F}\ Dec, Triv}{[\mathbf{A} \Leftrightarrow \mathbf{B}]^F}\ Equiv$$

Hence $\Phi_{cl} * [\neg(\mathbf{A} = \mathbf{B})]^T \vdash_{\mathcal{ER}_{fc}} \Phi_{cl} * [\neg(\mathbf{A} = \mathbf{B})]^T * [\mathbf{A} \Leftrightarrow \mathbf{B})]^F$ and we get the conclusion as a consequence of lemma 3.15(2).

∇_q We show that if $\Phi_{cl} * [\neg(\mathbf{F} \doteq^{\alpha\to\beta} \mathbf{G})]^T * [\neg(\mathbf{F}\ w \doteq^{\beta} \mathbf{G}\ w)]^T \vdash_{\mathcal{ER}_{fc}} \Box$, then $\Phi_{cl} * [\neg(\mathbf{F} \doteq \mathbf{G}))]^T \vdash_{\mathcal{ER}_{fc}} \Box$. Note that $\Phi_{cl} * [\neg(\mathbf{F} \doteq \mathbf{G})]^T * [\neg(\mathbf{F}\ w \doteq \mathbf{G}\ w)]^T \equiv \Phi_{cl} * [\neg\Pi(\lambda Q.\ \neg(Q\ \mathbf{F}) \vee (Q\ \mathbf{G}))]^T * [\neg\Pi(\lambda P.\ \neg(P\ (\mathbf{F}\ w)) \vee (P\ (\mathbf{G}\ w)))]^T \vdash_{\mathcal{CNF}} \Phi_{cl} * [q\ \mathbf{F}]^T * [q\ \mathbf{G}]^F * [p\ (\mathbf{F}\ w)]^T * [p\ (\mathbf{G}\ w)]^F$ and that $\Phi_{cl} * [\neg(\mathbf{F} \doteq \mathbf{G})] \vdash_{\mathcal{CNF}} \Phi_{cl} * [r\ \mathbf{F}]^T * [r\ \mathbf{G}]^F$, where $p_{\beta\to o}, q_{(\alpha\to\beta)\to o}$ and $r_{(\alpha\to\beta)\to o}$ are new Skolem constants. Now consider the following derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[r\ \mathbf{F}]^T \quad [r\ \mathbf{G}]^F}{[r\ \mathbf{F} = r\ \mathbf{G}]^F}\ Res}{[\mathbf{F} = \mathbf{G}]^F}\ Dec, Triv}{[\mathbf{F}\ s = \mathbf{G}\ s]^F}\ Func}{[t\ (\mathbf{F}\ s)]^T}\ Leib}{[t\ (\mathbf{G}\ s)]^F}$$

Here again s_α and $t_{\beta\to o}$ are new Skolem constants. Hence $\Phi_{cl} * [r\ \mathbf{F}]^T * [r\ \mathbf{G}]^F \vdash_{\mathcal{ER}_{fc}} \Phi_{cl} * [r\ \mathbf{F}]^T * [r\ \mathbf{G}]^F * [t\ (\mathbf{F}\ s)]^T * [t\ (\mathbf{G}\ s)]^F$. Now the conclusion follows from the assumption as s, t and r are only renamings of the Skolem symbols w, p and q and as all do not occur in Φ_{cl}.

To see that Γ_Σ is saturated let $\mathbf{A} \in wff_o(\Sigma)$ and $\Phi \subseteq cwff_o(\Sigma)$ with $\Phi_{cl} \not\vdash_{\mathcal{ER}_{fc}} \Box$. We have to show that $\Phi_{cl} * \mathbf{A} \not\vdash_{\mathcal{ER}_{fc}} \Box$ or $\Phi_{cl} * \neg\mathbf{A} \not\vdash_{\mathcal{ER}_{fc}} \Box$. For that suppose $\Phi_{cl} \not\vdash_{\mathcal{ER}_{fc}} \Box$, but $\Phi_{cl} * \mathbf{A} \vdash_{\mathcal{ER}_{fc}} \Box$ and $\Phi_{cl} * \neg\mathbf{A} \vdash_{\mathcal{ER}_{fc}} \Box$. By lemma 3.15(1) we get that $\Phi_{cl} * \mathbf{A} \vee \neg\mathbf{A} \vdash_{\mathcal{ER}_{fc}} \Box$, and hence, since $\mathbf{A} \vee \neg\mathbf{A}$ is a tautology, it must be the case that $\Phi_{cl} \vdash_{\mathcal{ER}_{fc}} \Box$, which contradicts our assumption. □

Remark 3.17 ***(Eager Unification).*** Different to Huet [Hue72, Hue73a] eager unification is essential within our approach. This is illustrated by the argumentations for ∇_b and ∇_q in the completeness proof 3.16 as well as most of the examples presented in section 7.

Remark 3.18 ***(Restricted rule** Leib).* Even though rule *Leib* is employed to arbitrary types in the completeness proof (see 5.20(∇_q)) the author claims that this rule can be restricted to unification constraints between terms of primitive type:

$$\dfrac{\mathbf{C} \vee [\mathbf{M}_\alpha = \mathbf{N}_\alpha]^F \quad \alpha \in \{o, \iota\}}{\mathbf{C} \vee [\forall P_{\alpha\to o}.\ P\ M \Rightarrow P\ N]^F}\ Leib'$$

Unfortunately a formal proof for this conjecture which is of great practical importance is still missing. It would be enough to prove that if $\Phi_{cl} * [\mathbf{A} \doteq^{\alpha\to\beta} \mathbf{B})]^F \vdash_{\mathcal{ER}_{fc}} \Box$ then $\Phi_{cl} * [\mathbf{A}\ s \doteq^{\beta} \mathbf{B}\ s]^F \vdash_{\mathcal{ER}_{fc}} \Box$ for any new Skolem term s_α. Note that $[\forall X_\alpha.\ \mathbf{A}X \doteq^{\alpha} \mathbf{B}X] \vdash_{\mathcal{CNF}} [\mathbf{A}\ s \doteq^{\beta} \mathbf{B}\ s]^F$. Thus, this lemma expresses one direction of the functional extensionality property formulated with Leibniz equality and with this lemma we could reduce the applications of rule *Leib* in all completeness proofs discussed in this paper to the restricted rule *Leib'*. This result would be of practical importance as it allows to restrict the search space extremely. When we later extend our extensional higher-order resolution into a corresponding RUE-resolution approach it may even be the case that rule *Leib* generally becomes admissible (c.f. remark 6.1)

3.5 Theorem Equivalence

With respect to our theoretical goal of proving Henkin completeness it does not make a difference whether exhaustive clause normalisation derivations are grouped together in only one rule *Cnf* like in calculi $\mathcal{ER}_f$ and $\mathcal{ER}$, or if the single clause normalisation rules are lifted as single inference rules

to calculus level as in calculus $\mathcal{ER}_{fc}$. But note that there is a practical motivation as well: calculus $\mathcal{ER}_{fc}$ allows to avoid redundant applications of identical unification derivations to all proper clauses belonging to the same abstract clause. For instance, it may be more appropriate first to apply unification rules to a non-proper clause and then to apply clause normalisation rule to the result than the other way around.

In the following lemma we make use of the following two important conventions for this paper:

Remark 3.19 ***(Convention).***

1. Whereas the $\mathcal{CNF}$ and the $\mathcal{UNI}$ rules aside from *Subst* do not directly influence each others applicability we have to admit that they may indirectly influence each others applicability via Skolemisation. More precisely if r_2 is rule Π^T and r_1 is rule *Func* the arity of the Skolem term that is introduced in *Func* increases its arity when switching r_1 and r_2. We have already pointed out in remark 3.1 that we ignore this fact here.

2. Another important and simplifying convention in all proof transformations in this paper is that we consider proper proof trees instead of proof graphs such that each derived clause in a derivation is used exactly once as a premise clause in one of the following derivation steps. An important and simplifying convention in all proof transformations in this paper is that we consider proper proof trees instead of proof graphs such that each derived clause in a derivation is used exactly once as a premise clause in one of the following derivation steps.

Lemma 3.20 (Derivability of Proper Clauses). *For each proper clause $\mathcal{C}$ and clause set Φ such that $\Delta_1 : \Phi \vdash_{\mathcal{ER}_{fc}} \mathcal{C}$ we have that there is a $\mathcal{ER}_f$-derivation $\Delta_2 : \Phi \vdash_{\mathcal{ER}_f} \mathcal{C}$.*

Proof: The proof idea is to show that the single, distributed clause normalisation steps can be grouped in exhaustive $\mathcal{CNF}$-chains again which can then be replaced by rule *Cnf*. The proof is by induction on the length n of derivation Δ_1 and the base case ($n \equiv 0$) is trivial. In the induction step ($n > 0$) we consider the first step in derivation $\Delta_1 : \Phi * \vdash^{r_1} \Phi * \mathcal{D}_1 \vdash_{\mathcal{ER}_{fc}} \mathcal{C}$ where $r_1 \in \mathcal{ER}_{fc}$ and proceed by examining all possibilities for r_1:

$r_1 \in \{Res, Fac, Prim\}$ As these rules operate only on proper clauses we have on the one hand that the first step is also a proper $\mathcal{ER}_f$ derivation step. On the other hand we know by induction hypothesis that there is a proper $\mathcal{ER}_f$ derivation $\Phi * \mathcal{D}_1 \vdash_{\mathcal{ER}_f} \mathcal{C}$. Thus, $\Phi \vdash_{\mathcal{ER}_f} \mathcal{C}$.

$r_1 \in \mathcal{UNI}_f$ Analogous to above the first step is also a proper $\mathcal{ER}_f$ derivation step and thus the assertion follows immediately by induction hypotheses.

$r_1 \in \mathcal{CNF}$ In this case we look for the first step in derivation Δ_1 that employs a non $\mathcal{CNF}$-rule. Without loss of generality let us assume that this happens in step n for an $n > 1$. Then Δ_1 has form $\Delta_1 : \Phi \vdash^{r_1} \Phi * \mathcal{D}_1 \vdash^{r_2} \ldots \vdash^{r_{n-1}} \Phi * \mathcal{D}_1 * \ldots * \mathcal{D}_{n-1} \vdash^{r_n} \Phi * \mathcal{D}_1 * \ldots * \mathcal{D}_n \vdash_{\mathcal{ER}_{fc}} \mathcal{C}$ such that $r_j \in \mathcal{CNF}$ for $1 < j < n$. Without loss of generality let us assume that each of the steps r_j employ the freshly generated clause from the previous step as premise clause (we can reorder the derivation steps in Δ_1 without any influence to the particularly derived clauses such that this assumption is met).
If $r_n \in \{Res, Fac, Prim\}$ $\mathcal{D}_n$ must be a proper clause, such that we can obviously replace the initial $n-1$ derivation steps in Δ_1 by a single application of rule *Cnf* in calculus $\mathcal{ER}_f$. Now we again employ the convention that we consider proper proof trees instead of proof graphs in our formal proofs such that each derived clause in a derivation is used exactly once as a premise clause in one of the following derivation steps. Consequently $\Delta_1 : \Phi \vdash_{\mathcal{CNF}} \Phi * \mathcal{D}_n \vdash_{\mathcal{ER}_{fc}} \mathcal{C}$. Now the assertion follows by induction hypotheses.
If $r_n \in \mathcal{UNI}_f$ we verify that the clause normalisation steps r_j for $1 \leq j < n$ do definitely not affect the unification constraints of the involved clauses. Thus we can obviously apply rule r_n in the first place in Δ_1 as well. Furthermore note that the clause normalisation steps r_j for $1 \leq j < n$ are applicable to the result of this new first step in Δ_1 (note that if $r_n \equiv Subst$, then this step does not introduce a non-proper literal) and that the result of this derivation chain in the n-th step is clause $\mathcal{D}_n$. Now the assertion follows again by induction hypotheses.

□

As □ is also a proper clause we immediately get the following corollary.

Corollary 3.21 (Theorem Equivalence of $\mathcal{ER}_{fc}$ and $\mathcal{ER}_f$). *The calculi $\mathcal{ER}_{fc}$ and $\mathcal{ER}_f$ are theorem equivalent, i.e. $\Phi \vdash_{\mathcal{ER}_{fc}} \square$, iff $\Phi \vdash_{\mathcal{ER}_f} \square$.*

Whereas we already hinted that in our extensional higher-order resolution approach it is no longer possible to delay all (pre-)unification rules our claim will be that the additional rule *FlexFlex* can still be delayed until the end of a refutation. This will give us that rule *FlexFlex* is not needed at all as □ is defined modulo *flex-flex* constraints.

Conjecture 3.22 (Theorem Equivalence of $\mathcal{ER}$ and $\mathcal{ER}_{fc}$ (or $\mathcal{ER}_f$)). *The calculi $\mathcal{ER}$ and $\mathcal{ER}_{fc}$ (or $\mathcal{ER}_f$) are theorem equivalent.*

Even though a formal proof for this important conjecture has not yet been carried out formally there is strong evidence given for its validity. None of the challenging examples discussed in this or in any of the papers [BK98a, BK98b, Ben97, BK97a] needs rule *FlexFlex*. Furthermore, none of the more than 200 with respect to extensionality very interesting examples from the MIZAR-articles *Boolean and Basic Properties of Sets* [TS89, Byl89] needs to employ the *FlexFlex* rule within our case study with the extensional higher-order resolution prover LEO [BK98b]. And strong evidence is given by the direct proof carried out in [BK98a] (which admittedly lacks a bit of transparency within the lifting lemma).

4 Primitive Equality

In the extensional higher-order resolution approach discussed in the previous section equality is treated as a concept defined by Leibniz' principle. While this way of treating equality is theoretically a convenient and suitable solution it turns out that this solution is rather inappropriate in practice[12] as Leibniz equality introduces new flexible literals into the search space and thereby increases the amount of blind search with primitive resolution rule *Prim*. Furthermore proofs for problem formulations that employ Leibniz equality are rather unintuitive and hardly to understand for non-experts. Illustrating examples for such unintuitive problem formulations and proofs when using Leibniz equality are **E4** and **E5** as discussed in [BK98a].

Thus our aim is to avoid Leibniz definition of equality and to use primitive equality instead. Very important for the further discussion in this section is that unlike in first-order logic we do not have the choice in higher-order to consider equality as either defined or as primitive. Instead we are generally forced to consider a mixed (primitive and defined) treatment of equality. The reason is that when assuming Henkin semantics infinitely many definitions of equality are implicitly provided by our higher-order language and there is no way to get rid of them. A definition of equality that is different from Leibniz equality is, for instance, discussed in [And86] at page 155. Furthermore, this paper provides illustrating examples in subsection 7.2. And aside from all the sensible definitions of equality in higher-order logic there is always ways to introduce additional artificial ones. This can, for instance, be done by adding arbitrary tautologous sub-formulas to a definition of equality. For example, if $\mathbf{A}^1 \ldots \mathbf{A}^n$ and $\mathbf{B}^1 \ldots \mathbf{B}^n$ are tautologous sentences then we can define equality based on Leibniz equality also by $=^\alpha := \lambda X_\alpha . \lambda Y_\alpha . \forall P_{\alpha\to o} . (\mathbf{A}^1 \wedge \ldots \wedge \mathbf{A}^n \wedge P\ X) \Rightarrow (P\ Y \wedge \mathbf{A}^1 \wedge \mathbf{A}_1 \wedge \ldots \wedge \mathbf{A}^n$. It is obvious that by this trick we can introduce infinitely many valid but different definitions of equality. And as it is undecidable whether a formula is a tautology we also get that it is undecidable if an expression is equivalent to Leibniz equality and describes the intended equality relation. This argumentation immediately gives us the following corollary:

Corollary 4.1 (Defined equality in higher-order logic). *Given a higher-order sentence* **A**. *There is no procedure that can decide if* **A** *contains a sub-formula that expresses the equality between two terms.*

[12] Even though the already mentioned successful case study with the LEO-system on the the examples from the MIZAR-articles Boolean and Basic Properties of sets showed that Leibniz equality is sufficient for proving simple problems about sets.

Paramodulation: (defined for proper clauses only)

$$\frac{[\mathbf{A}[\mathbf{T}_\beta]]^\alpha \vee C \quad [\mathbf{L} =^\beta \mathbf{R}]^T \vee D}{[\mathbf{A}[\mathbf{R}]]^\alpha \vee C \vee D \vee [\mathbf{T} =^\beta \mathbf{L}]^F}\ Para$$

$$\frac{[\mathbf{A}]^\alpha \vee C \quad [\mathbf{L} =^\beta \mathbf{R}]^T \vee D}{[P_{\alpha\to o}\mathbf{R}]^\alpha \vee C \vee D \vee [\mathbf{A} =^o P_{\beta\to o}\mathbf{L}]^F}\ Para'$$

Figure 4: Adapted paramodulation rule *Para* and a higher-order specific reformulation *Para'*

A consequence of this lemma is that we generally have to assume the presence of some defined equations in input problems passed to our calculus which we can not detect and remove by primitive equations. Thus, even if we add a primitive equality treatment to our calculus we still have to ensure that our calculus – like the calculus $\mathcal{ER}$ discussed in the last section — can handle Leibniz equality as well as all alternative definitions of equality.[13]. This is different to the first-order case, where one has the choice to define equality (e.g. by axiomatising equality as a congruence relation) or to consider it as a primitive notion and to add special equality rules (e.g. paramodulation rules) to the calculus. Thus in our setting we first have to leave the calculus $\mathcal{ER}$ as it is and we can not – as one might wish to – remove the extensionality rules from the beginning. Sure, when adding some special primitive equality rules we can then examine if some of the given rules become admissible and hence superfluous (c.f. remark 6.1).

We will now focus on the first-order paramodulation approach and try to adapt it to our higher-order setting. The anticipated result of this attempt is that just adapting the first-order paramodulation rules is not sufficient to ensure Henkin completeness. Again the problem is caused by the extensionality properties and as will see additional extensionality rules are needed — but this time with respect to positive equations. Unfortunately the resulting higher-order paramodulation calculus seems to be inappropriate within specific domains — e.g. for examples about sets, when sets are coded as characteristic functions and where extensionality properties play an important role — as it looses its pure term-rewriting character.

And aside from the problem that well founded reduction orderings are hardly to define in higher-order logic the higher-order term-rewriting idea has to face a second problem: it seems to be rather complicate to guide an approach with a mixed term-rewriting and difference-reducing character in practice. Thus, we will later develop a second calculus that adapts the ideas of first-order RUE-resolution [Dig79]. The latter extensional higher-order RUE-resolution approach has a pure difference reducing character which might be easier to guide in practice.

5 Extensional Higher-Order Paramodulation: $\mathcal{EP}$

5.1 A Naive and Incomplete Adaption of Paramodulation

Figure 4 displays the traditional paramodulation rule and introduces a more elegant higher-order reformulation[14] which is simply motivated by the corresponding resolution step with the clause obtained by interpreting the first literal as a positive Leibniz equation.

Note that we assume the symmetric case for both rules. The adapted traditional paramodulation rule *Para* allows to replace an arbitrary subterm $\mathbf{T}_\beta$ by $\mathbf{R}_\beta$ provided that $\mathbf{T}$ and $\mathbf{L}$ are

[13] We want to point out that this argumentation even holds for the weaker semantical notion of functional Σ-models with property $\mathfrak{q}$ ($\mathfrak{M}_\mathfrak{q}$'s) as introduced in [BK97b]. Even though this semantical notion is weaker than Henkin semantics equality can still be defined.

[14] This rule was suggested by Michael Kohlhase in a private conversation.

unifiable. Analogous to higher-order resolution and factorisation rules unification has to be delayed and thus a unification constraint $[\mathbf{T} = \mathbf{L}]^F$ is added to the resulting clause. Unfortunately we thereby introduce with each application of rule *Para* twice[15] as many new clauses into the search space as subterms $\mathbf{T}_\alpha$ of type α are given in the term $\mathbf{A}$. Consider for example the unit clauses *c1:* $[p\ (f\ (f\ a))]^T$ and $\mathcal{C}_2 : [f = h]^T$, where $p_{\iota\to o}, f_{\iota\to\iota}, h_{\iota\to\iota}$ and a_ι are constants. By application of paramodulation in left to right direction we obtain the follwing two clauses: $\mathcal{C}_3 : [p\ (h\ (f\ a))]^T \vee [f = f]^F$ and *c4:* $[p\ (f\ (h\ a))]^T \vee [f = f]^F$. The unification constraints can be immediately eliminated with rule *Triv* here.

Rule *Para'* combines the expressivness of our higher-order language with the power of higher-order (pre-)unification and avoids to introduce too many clauses into the search space but instead introduces only one new clause with a new flexible literal head. On can easily convince himself that this rule has a very simple motivation: It simply describes the clause that we obtain from the right premise when replacing the primitive equality with Leibnizequality and applying clause normalization: $[P\ \mathbf{L}]^F \vee [P\ \mathbf{R}]^T \vee D$ (or by a additional primitive substitution step: $[P\ \mathbf{L}]^T \vee [P\ \mathbf{R}]^F \vee D$.).

When applying rule *Para'* to the clauses $\mathcal{C}_1$ and $\mathcal{C}_2$ above from left to right we introduce only one new clause: $\mathcal{C}_5 : [P\ \lambda X_\iota.\ a]^T \vee [p\ (f\ (f\ a)) = P\ f]^F$. But note that by eager pre-unification we can generate the following four instantiations for P and propagate this partial solutions with rule *Subst* to the literal $[P\ \lambda X_\iota.\ a]^T$:

1) $P \leftarrow \lambda Z_{\iota\to\iota}.\ p\ (f\ (f\ a))$ 2) $P \leftarrow \lambda Z_{\iota\to\iota}.\ p\ (Z\ (f\ a))$
3) $P \leftarrow \lambda Z_{\iota\to\iota}.\ p\ (f\ (Z\ a))$ 4) $P \leftarrow \lambda Z_{\iota\to\iota}.\ p\ (Z\ (Z\ a))$

The pure imitation solution (1) introduces clause $\mathcal{C}_1$ again and is thus redundant.[16] By back-propagation of solutions (2) and (3) we obtain exactly the clauses $\mathcal{C}_3$ and $\mathcal{C}_4$, which are the results of traditional paramodulation rule. Solution (4) is the most interesting one as it encodes the simultaneous application of $[f = h]^T$ to both subterms f in $[p\ (f\ (f\ a))]^T$ with the traditional rule *Para*. Thus, the only clause resulting from the application of rule *Para'* encodes all the possible traditional paramodulation steps together with all simultaneous applications and even the original clause itself. The only problem for practical applications is that we introduce a new flexible head with rule *Para'* and hence primitive substitution rule becomes applicable. As this rather less useful, a probably good heuristic in practice is to avoid such primitive substitution steps on flexible heads generated by rule *Para'*.

Remark 5.1 ***(Reflexivity Resolution).*** We want to point out that reflexivity resolution is already embedded into the calculus $\mathcal{ER}$. The reader can easily convice himself that reflexivity resolution is derivable with the help of the unification rules. The unification rules already operate on negated equality literals – also called unification constraints – and thus each negative equation between two unifiable terms are tackled by them.

Remark 5.2 ***(Paramodulation into Unification Constraints).*** It turns out that paramodulation on unification constraints is not neccessary and in fact we can show derivability.

$$\frac{\mathcal{C}_1 : C \vee [\mathbf{A}[\mathbf{T}_\alpha]\ = \mathbf{B}]^F \quad \mathcal{C}_2 : [\mathbf{L} =^\alpha \mathbf{R}]^T \vee D}{\mathcal{C}_3 : C \vee D \vee [\mathbf{A}[\mathbf{R}_\alpha]\ = \mathbf{B}]^F \vee [\mathbf{T} =^\alpha \mathbf{L}]^F}\ Para$$

Each such step that employs paramodulation on unification constraints can be replaced by the following derivation (p is a Skolem constant):

$Leib(\mathcal{C}_1), \mathcal{CNF}$:	$\mathcal{D}_1 : C \vee [p\ \mathbf{A}[\mathbf{T}_\alpha]^T$
	$\mathcal{D}_2 : C \vee [p\ \mathbf{B}]^F$
$Para(\mathcal{D}_1, \mathcal{C}_2)$:	$\mathcal{D}_3 : C \vee D \vee [p\ \mathbf{A}[\mathbf{R}_\alpha]]^T \vee [\mathbf{T} =^\alpha \mathbf{L}]^F$
$Res(\mathcal{D}_3, \mathcal{D}_2), Fac, Triv$:	$\mathcal{D}_4 : C \vee D \vee [(p\ \mathbf{A}[\mathbf{R}_\alpha]) = (p\ \mathbf{B})]^F \vee [\mathbf{T} =^\alpha \mathbf{L}]^F$
$Dec(\mathcal{D}_4), Triv$:	$\mathcal{D}_5 : C \vee D \vee [\mathbf{A}[\mathbf{R}_\alpha] = \mathbf{B}]^F \vee [\mathbf{T} =^\alpha \mathbf{L}]^F$

[15] Because of symmetry.

[16] On the other hand this possibly leads to interesting heuristics in practice: When applying rule *Para'* we can remove the left premise clause from the search space, as this clause gets encoded into the result of the paramodulation step.

On the one hand paramodulation into unification constraints shortens proofs and in some examples this seems to be very appropriate, but on the other hand this may be hard to guide in practice.

Definition 5.3 ($\mathcal{EP}_{naive}$). The calculus $\mathcal{EP}_{naive}$ consists of the rules of calculus $\mathcal{ER}$ (see figure 2) enriched by the paramodulation rule *Para* (see figure 4). We assume that the result of each rules application is transformed into head-normal form. A set of formulas Φ is refutable in calculus $\mathcal{EP}_{naive}$ iff there is a derivation $\Delta : \Phi_{cl} \vdash_{\mathcal{EP}_{naive}} \square$, where $\Phi_{cl} := \{[\mathbf{F}_{\downarrow_h}]^T | \mathbf{F} \in \Phi\}$ is the set of clauses obtained from Φ by simple pre-clausification. We want to point out that primitive equations are not expanded by Leibniz definition.

Next, we discuss soundness of the extended calculus $\mathcal{EP}_{naive}$ and show by a counterexample that $\mathcal{EP}_{naive}$ is not complete with respect to Henkin semantics.

Theorem 5.4 (Soundness of $\mathcal{EP}_{naive}$). *The calculus $\mathcal{EP}_{naive}$ is sound with respect to standard semantics.*

Proof: Soundness of the traditional paramodulation rule *Para* is obvious: Given a standard model M for the two premise clauses then one can easily show that the paramodulant is also valid in M as either the unification constraint evaluates to $\bot$ and we are done or it evaluates to $\top$ giving reason to the validity of literal $[\mathbf{A}[\mathbf{R}]]]^\alpha$ in case $[\mathbf{A}[\mathbf{T}]]]^\alpha$ guaranteed the validity of the first premise clause and $[\mathbf{L} = \mathbf{R}]^T$ the validity of the second premise (all other cases are trivial).

The proof of soundness for the new rule *Para'* is analogous, even though this rules looks more complicate: We consider all possible variable assignments φ that map variable $P_{\alpha\to o}$ to a function in the domain $D_{\alpha\to o}$ and employ an analogous argumentation like above. □

Theorem 5.5 (Incompleteness of $\mathcal{EP}_{naive}$). *The calculus $\mathcal{EP}_{naive}$ is incomplete with respect to Henkin semantics.*

Proof: The assertion is proven by the following counterexamples to the assumption of Henkin completeness of calculus $\mathcal{EP}_{naive}$:

*Example 5.6 (**Incompleteness of Paramodulation**).*

$\mathbf{E}_1^{Para}$ $\neg\exists X_o . (X = \neg X)$
This formula expresses that the negation operator is fix-point free which is obviously the case in Henkin semantics. Our calculus is not able to find a proof as clause normalisation of the negated assertion leads to the single clause

$$\mathcal{C}_1 : [a = \neg a]^T$$

where a_o is a new Skolem constant. The only rule that is applicable is self-paramodulation on positions $\langle 1\rangle$, $\langle 2\rangle$ and $\langle\rangle$ leading to the following clauses:

$$\begin{array}{llll} Para(\mathcal{C}_1,\mathcal{C}_1)\ at\ \langle 1\rangle : & \mathcal{C}_2 : [a = \neg a]^T \vee [\neg a = a]^F & \mathcal{C}_3 : [\neg a = \neg a]^T \vee [a = a]^F \\ Para(\mathcal{C}_1,\mathcal{C}_1)\ at\ \langle 2\rangle : & \mathcal{C}_4 : [a = \neg a]^T \vee [a = \neg a]^F & \mathcal{C}_5 : [a = a]^T \vee [\neg a = a]^F \\ Para(\mathcal{C}_1,\mathcal{C}_1)\ at\ \langle\rangle : & \mathcal{C}_6 : [a]^T \vee [\neg a = (a = \neg a)]^F & \mathcal{C}_7 : [a]^F \vee [a = (a = \neg a)]^F \end{array}$$

Case distinction on the possible denotations $\{\top, \bot\}$ for a shows that all these clauses are tautologies. Thus no refutation is possible in $\mathcal{EP}_{fc}$.

$\mathbf{E}_2^{Para}$ $\neg\exists G_{\iota\to\iota\to o} . \forall P_{\iota\to o} . \exists X_\iota . G\ X =^{\iota\to o} P$
This is a simple formulation of cantor's theorem stating that there exists no surjective function from the set of individuals into the set of sets of individuals. Clause normalisation results in

$$\mathcal{C}_1 : [G\ X =^{\iota\to o} p]^T$$

where $p_{\iota\to o}$ is a Skolem constant. Analogous to above, the positive extensionality properties a refutation is not possible.

$\mathbf{E}_3^{Para}$ $\exists M_{o\to o}.\ M \neq \emptyset$ where $\emptyset_{o\to o}$ is defined as $\lambda X_o.\ \bot$. This formula expresses that there exists a set of Booleans that is not empty. As the set of Booleans is fixed in Henkin semantics and has exactly two elements this assertion is obviously valid. Clause normalisation results in

$$\mathcal{C}_1 : [M = \lambda X_o.\ \bot]^T$$

where $M_{o\to o}$ is a free variable. Analogous to $\mathbf{E}_1^{Para}$ no refutation is possible.

$\mathbf{E}_4^{Para}$ $\neg\exists M_{o\to o}.\ M = \bar{M}$ where the set complement operator $\bar{}_{(o\to o)\to(o\to o)}$ is defined by $\lambda S_{o\to o}.\ \lambda X_o.\ \neg(X \in Y)$ and $\in_{o\to(o\to o)\to o}$ is just function (predicate) application $\lambda X_o.\ \lambda S_{o\to o}.\ SX$. This formula expresses that there is no set M of Booleans in Henkin semantics such that M and its complementary set $\overline{M}$ are identical. Clause normalisation leads to

$$\mathcal{C}_1 : [m = \lambda X_o.\ \neg(mX)]^T$$

where $m_{o\to o}$ is a fresh Skolem constant for M. In contrast to $\mathbf{E}_3^{Para}$ there is no free variable. Again no refutation is possible.

$\mathbf{E}_5^{Para}$ If $\forall P_{(\iota\to\iota\to o)\to(\iota\to\iota\to\iota\to o)}.\ (P\ q_{\iota\to\iota\to o} = P\ r_{\iota\to\iota\to o})$ Then $(\forall X_\iota.\ q\ X \neq \lambda Z_\iota.\ \neg(r\ X\ Z))$. The first equation expresses that all relations of type $\iota\to\iota\to\iota\to o$ that can be defined based upon relation q or r are equal (which in fact means that q and r must be equal). The second inequation in some sense says that q is the complement set of r, but uses an artificially complicated form (we employed the functional extensionality property once to $q \neq \lambda V_i, Z_i.\ \neg(r\ V\ Z)$). This assertion is again valid as in Henkin semantics the set of truth values contains exactly two elements. Our problem normalises to

$$\mathcal{C}_1 : [P\ q =^{\iota\to\iota\to\iota\to o} P\ r]^T \qquad \mathcal{C}_2 : [p\ X =^{\iota\to o} \lambda Z_\iota.\ \neg(r\ X\ Z)]^T$$

where $P_{\iota\to\iota\to o)\to(\iota\to\iota\to\iota\to o)}, X_\iota$ are free variables and $q_{\iota\to\iota\to o}, r_{\iota\to\iota\to o}$ are function constants. Note that aside from self-paramodulation no rule is applicable. Especially there is no paramodulation step possible between the clauses $\mathcal{C}_1$ and $\mathcal{C}_2$ as there is no subterm of type $\iota\to o$ in $\mathcal{C}_1$ and no subterm of type $\iota\to\iota\to\iota\to o$ in $\mathcal{C}_2$ such that a type conform term-rewriting becomes possible.

□

The overall problem in the first three examples is that our calculus provides no mechanism to detect positive equations with an implicitly embedded contradiction. For example the clause $[a_o = \neg a]^T$ is is implicitly contradictory with respect to the boolean extensionality principle. Examples $\mathbf{E}_2^{Para}$ and $\mathbf{E}_4^{Para}$ show that also functional extensionality is involved as the implicit contradiction follows here only with respect to both extensionality principles. The general problem is that in higher-order logic (when considering Henkin semantics as well as some weaker notions like the Σ-models in [BK97b] or Andrews $\mathcal{V}$-complexes [And71]) infinitely many semantical domains contain a fix point free function such as the negation operator or the set complement operator on a non-empty domain such as the set of truth values or the domain of all functions from truth values to truth values, etc.

In example $\mathbf{E}_5^{Para}$ none of the positive equality literals is contradictory by itself but only in connection with the other one. Here again a refutation is not possible in calculus $\mathcal{ER}_{fc}$. Here we need to employ the extensionality properties to the positive equation in order to introduce a positive equation of appropriate type, thereby making paramodulation rule applicable.

Remark 5.7 ***(Fix-point free functions).*** Given a functional type $\alpha\to o$. The semantical domain $\mathbf{Dom}_{(\alpha\to o)\to(\alpha\to o)}$ contains a fix-point free function provided that it is not empty (which must be the case in Henkin Semantics, as at least the identity function, i.e. the evaluation of $\lambda P_{\alpha\to o}.\ P_{\alpha\to o}$, must be an element of this domain). For instance, for $\mathbf{Dom}_{((\iota\to\iota)\to o)\to((\iota\to\iota)\to o)}$ we can choose the set complement operator defined with the help of the negation operator: $\lambda S_{(\iota\to\iota)\to o}.\ \lambda F_{\iota\to\iota}.\ \neg(S\ F)$.

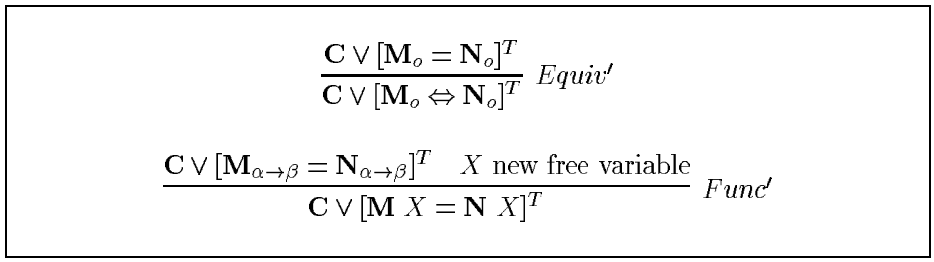


Figure 5: Positive extensionality treatment.

5.2 Positive Extensionality Rules

Before stepping further in the examination of our extensionality problem on positive equation literals let us first reconsider the analogous problem for negative equations which is already solved in our calculus. For example $[a_o = \neg\neg a]^F$ is an negative equation which is implicitly contradictory with respect to boolean extensionality. Our calculus already provides a solution to this problem as it interprets negative equation literals automatically as unification constraints for which an appropriate extensionality treatment is already available by the extensionality rules *Leib, Func* and *Equiv*. We already pointed out that reflexivity resolution is therefore superfluous. But even better, we do not need additional extensionality rules for negative equation literals here.

Unfortunately we do have to face the lack of extensionality principles in case of positive equation literals and as we have seen in the examples above, our calculus so far does not provide a solution to it. What we need is a way to test the inequality of the two hand sides of a positive equation with respect to the extensionality principles and also with respect to the knowledge provided by the other clauses in the search space (see for example $\mathbf{E}_5^{Para}$). This suggests to introduce analogous extensionality rules to *Func, Equiv* and *Leib* but now for positive equations as suggested in figure 5. It turns out that a positive counterpart to rule *Leib* is not needed:

*Remark 5.8 (**Rule** **Leib'**).* A positive counterpart for rule *Leib* would have the following form:

$$\frac{\mathbf{C} \vee [\mathbf{M}_\alpha = \mathbf{N}_\alpha]^T \quad \alpha \in \{o, \iota\}}{\mathbf{C} \vee [\forall P_{\alpha\to o}.\ P\ M \Rightarrow P\ N]^T}\ \mathit{Leib'}$$

This rule is not needed in our motivating examples and the completeness proof below. Thus, it should be possible to prove that rule *Leib'* is admissible for calculus $\mathcal{EP}$.

The reader is probably interested now in the concrete proofs of our challenging examples within our higher-order paramodulation calculus enriched by the positive extensionality rules. This refutations are presented in detail in subsection 7.3.

5.3 Basic Definitions

Analogous to section 3 we first further extend our extensional higher-order paramodulation approach by adding rule *FlexFlex* and lifting the single clause normalisation rules to calculus level instead of grouping exhaustive clause normalisation derivations together with rule *Cnf*.

The definitions for clause normalisation calculus $\mathcal{CNF}$ (see 3.2), $\mathcal{UNI}$ and $\mathcal{UNI}_f$ (see 3.4) need not to be modified and the lemmata 3.3 (Soundness of $\mathcal{CNF}$) and 3.5, 3.6 (properties of higher-order unification) will be employed in this section again.

Definition 5.9 (Extensional Higher-Order Paramodulation).

$\mathcal{EP}$ The calculus $\mathcal{EP}$ consists of the rules of calculus $\mathcal{ER}$ (see figure 2) enriched by the paramodulation rule *Para* (see figure 4) and the positive extensionality rules displayed in figure 5, i.e. $\mathcal{EP} := \mathcal{ER} \cup \{Para, Equiv', Func'\}$.

$\mathcal{EP}_f$ The extension $\mathcal{EP}_f$ of calculus $\mathcal{EP}$ that employs full higher-order unification instead of higher-order pre-unification is defined as $\mathcal{EP}_f := \mathcal{EP} \cup \{FlexFlex\}$.

$\mathcal{EP}_{fc}$ The calculus $\mathcal{EP}_{fc}$ that employs stepwise instead of exhaustive clause normalisation is defined by $\mathcal{EP}_{fc} := (\mathcal{EP}_f \backslash \{Cnf\}) \cup \mathcal{CNF}$.

For all calculi we assume that the result of each rule application is transformed into head-normal form. A set of formulas Φ is refutable in calculus $\mathcal{EP}$ iff there is a derivation $\Delta : \Phi_{cl} \vdash_{\mathcal{EP}} \square$ where $\Phi_{cl} := \{[\mathbf{F}_{\downarrow h}]^T | \mathbf{F} \in \Phi\}$ is the set obtained from Φ by simple pre-clausification. Unification literals are still only accessible to the unification rules.

Theorem 5.10 (Soundness of Extensional Higher-Order Paramodulation). *The calculi $\mathcal{EP}$, $\mathcal{EP}_f$ and $\mathcal{EP}_{fc}$ are sound with respect to standard semantics.*

Proof: We already know by 5.4 that calculus $\mathcal{EP}_{naive}$ is sound. The soundness of the new positive extensionality rules is obvious as they simply apply the extensionality properties which are valid in standard semantics. □

We will proceed analogous to section 3 and adapt lemma 3.12 which is hardly affected by the slight modifications to our calculus. Lemma 3.11 is not affected at all.

Lemma 5.11 (Proper Derivations). *For each non-proper clause $\mathcal{D}$, proper clause $\mathcal{C}$ and clause set Φ such that $\Phi * \mathcal{D} \vdash_{\mathcal{EP}_{fc}} \mathcal{C}$ we have that $\Phi \cup \mathcal{CNF}(\mathcal{D}) \vdash_{\mathcal{EP}_{fc}} \mathcal{C}$.*

Proof: Analogous to lemma 3.10. In the induction step the case where $r \equiv Para$ is analogous to the cases $r \in \{Res, Fac, Prim\}$ and the cases $r \in \{Func', Equiv'\}$ are analogous to the case $r \in \mathcal{UNI}_f$. □

5.4 Lifting Properties

The clause normalisation lifting property stated in lemma 3.11 is not affected by the modifications to the calculus and will be employed in the following lifting lemma for calculus $\mathcal{EP}_{fc}$. The new problem within this lemma is that we have to take care of the additional logical connectives $=^\alpha$ in the extended signature. But fortunately rule *Prim* got automatically extended as well and allows now to introduce all logical connectives $=^\alpha$ at head position analogous to the connectives $\neg, \vee, \Pi^\alpha$ so far.

Lemma 5.12 (Lifting Lemma for $\mathcal{EP}_{fc}$). *Let Φ be a set of clauses, $\mathcal{D}_1$ be a clause and σ a substitution. We have that:*

1. *For each derivation $\Delta_1 : \Phi_\sigma \vdash^1_{\mathcal{EP}_{fc}} \mathcal{D}_1$ there exists a substitution δ, a clause $\mathcal{D}_2$ and a derivation $\Delta_2 : \Phi \vdash_{\mathcal{EP}_{fc}} \mathcal{D}_2$ such that $(\mathcal{D}_2)_\delta \equiv \mathcal{D}_1$.*
2. *For each derivation $\Delta_1 : \Phi_\sigma \vdash_{\mathcal{EP}_{fc}} \mathcal{D}_1$ there exists a substitution δ, a clause $\mathcal{D}_2$ and a derivation $\Delta_2 : \Phi \vdash_{\mathcal{EP}_{fc}} \mathcal{D}_2$ such that $(\mathcal{D}_2)_\delta \equiv \mathcal{D}_1$.*

Proof:

(1) The proof is by case distinction on all rules in $\mathcal{EP}_{fc}$ and in all cases we will motivate that there is a derivation Δ_2 as required.

Res,Prim,Fac,Cnf,Subst The corresponding argumentations in 3.12 are not affected by the additional rules or the availability of primitive equality symbols $=^\alpha$ in the signature.

Triv, Func, Dec, FlexRigid, FlexFlex, Equiv, Leib This cases are affected by the new primitive equality symbols $= \alpha$ in the signature. More concretely we may have a unification constraint $[\mathbf{T}_1 = \mathbf{T}_2]^F$ on the instantiated level but a negative literal with a flexible head $[H\ \overline{\mathbf{U}^n}]^F$ on the abstract level. In order to enable the application of the particular unification on the abstract layer as well we can use primitive substitution rule *Prim* in connection with rule *Subst* in order to introduce the logical connective $=$ at head position leading to the abstract unification constraint $[H_1\ \overline{\mathbf{U}^n} = H_2\ \overline{\mathbf{U}^n}]^F$. The latter unification constraint is obviously more general as the instantiated one. The remaining argumentations are analogous to the corresponding ones discussed in 3.12.

Leib',Equiv',Para The argumentation is analogous to the above cases.

(2) The proof is by induction on the length of derivation Δ_1. The base case is trivial and in the induction step we first employ statement (1) and then the induction hypothesis. □

5.5 Completeness

Analogous to subsection 3.4 we analyse in this subsection Henkin completeness of calculus $\mathcal{EP}_{fc}$. The calculi $\mathcal{EP}_f$ and $\mathcal{EP}$ are then discussed in subsection 5.6. Like in the case of extensional higher-order resolution, theorem equivalence between $\mathcal{EP}$ and $\mathcal{EP}_{fc}$ has not been formally proven yet and will thus only be presented as a conjecture.

Before we present the completeness theorem 5.20 for $\mathcal{EP}_{fc}$ we first adapt the two lemmata 3.15 and 3.15. The first one compares refutability of clause sets in head-normal form with the refutability of clauses in $\beta\eta$-normal form and the second one discusses some important refutational properties.

Lemma 5.13 (Head-Normal Form). *Let Φ be a set of clauses. If $\Delta : \Phi_{\downarrow\beta\eta} \vdash_{\mathcal{EP}_{fc}} \square$, then $\Delta' : \Phi_{\downarrow_h} \vdash_{\mathcal{EP}_{fc}} \square$*

Proof: The proof is analogous to lemma 3.13. In the induction step we additionally have to consider the cases where $r \in \{Para, Func', Equiv'\}$ which are analogous to the cases where $r \in \{Prim\} \cup \mathcal{CNF}$. □

Lemma 5.14. *Let Φ be a set of clauses and $\mathbf{A}, \mathbf{B}$ be formulas. We have that:*

1. *If $\Phi * [\mathbf{A}]^T \vdash_{\mathcal{EP}_{fc}} \square$ and $\Phi * [\mathbf{B}]^T \vdash_{\mathcal{EP}_{fc}} \square$ then $\Phi * [\mathbf{A} \vee \mathbf{B}]^T \vdash_{\mathcal{EP}_{fc}} \square$.*
2. *If $\Phi * [\mathbf{A}]^T * [\mathbf{B}]^F \vdash_{\mathcal{EP}_{fc}} \square$ and $\Phi * [\mathbf{A}]^F * [\mathbf{B}]^T \vdash_{\mathcal{EP}_{fc}} \square$ then $\Phi * [\mathbf{A} \Leftrightarrow \mathbf{B}]^F$*

Proof: Analogous to lemma 3.15. The new rules do not affect the argumentations. □

In the completeness proof below we will employ a generalised paramodulation rule *GPara* that is allowed to operate also on non-proper clauses but which only rewrites identical terms and employs only unit equations instead of conditional equations.

Definition 5.15 (Generalised Paramodulation Rule). The generalised paramodulation rule *GPara* is defined as follows:

$$\frac{[\mathbf{T}[\mathbf{A}_\beta]]^\alpha \vee \mathbf{C} \quad [\mathbf{A} =^\beta \mathbf{B}]^T}{[\mathbf{T}[\mathbf{B}]]^\alpha \vee \mathbf{C}}\ GPara$$

This rule is not restricted to proper clauses and is furthermore allowed to operate on unification constraints. Thus *GPara* generalises rule *Para*. On the other hand *GPara* also restricts *Para* as unification is not employed and thus only identical terms can be replaced when applying this rule. Furthermore the second premise clause is assumed to consists only of a single positive equation literal.

We want to point out that rule *GPara* is specially designed for being employed in the completeness proof 5.20 to ensure abstract consistency property $\nabla_{\mathfrak{e}}(s)$, and the only reason we introduce the latter restriction for rule *GPara* is that we want to ease the proofs of the lemmata below as much as possible.

Definition 5.16 (Generalised Resolution Rules). The generalised resolution rules $GRes_1$, $GRes_2$ and $GRes_2$ are defined as follows:

$$\frac{[\mathbf{A}]^\alpha \vee C \quad [\mathbf{A}]^\beta \vee D \quad \alpha \neq \beta}{(C \vee D)}\ GRes_1$$

$$\frac{[\mathbf{A}\ \overline{Y^n}]^\alpha \vee C \quad [X\ \overline{\mathbf{T}^n}]^\beta \vee D \quad \alpha \neq \beta \text{ and } \overline{Y^n} \notin \mathbf{Free}(\overline{\mathbf{T}^n})}{(C \vee D)_{[\mathbf{A}/X, \overline{\mathbf{T}^n}/\overline{Y^n}]}}\ GRes_2$$

$$\frac{[\mathbf{A}_{\overline{\gamma}\to o}\ \overline{\mathbf{T}^n_\gamma}]^\alpha \vee C \quad [\mathbf{A}_{\overline{\gamma}\to o}\ \overline{X^n_\gamma}]^\beta \vee D \quad \alpha \neq \beta}{(C \vee D)_{[\overline{T^n}/\overline{X^n}]}}\ GRes_3$$

These rules are not restricted to proper clauses and in this sense they extend rule *Res*. But note that the new rules are defined within special contexts only and in this sense they also restrict the rule *Res*.

We will now show that these three generalised resolution rules as well as the generalised paramodulation rule are in a weak sense (modulo subsequent clause normalisation) derivable in calculus $\mathcal{EP}_{fc}$. The proof for the weak derivability of *GPara* thereby employs the generalised resolution rules $GRes_1$ and $GRes_3$. $GRes_2$ is only needed within for the within the proof of the weak derivability property for the generalised resolution rules itself.

Lemma 5.17 (Weak Derivability of Generalised Resolution Rules).

1. *Let* $\mathcal{C}_1 : [\mathbf{A}]^\alpha \vee \mathbf{C}$ *and* $\mathcal{C}_2 : [\mathbf{B}]^\beta \vee \mathbf{D}$ *be clauses such that there are clauses* $\mathcal{C}_3 : \mathbf{C}' \vee \mathbf{D}'$ *and* $\mathcal{C}_4$ *and a derivation* $\Delta_1 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash^r \mathcal{C}_3 \vdash_{\mathcal{CNF}} \mathcal{C}_4$ *where* $r \in \{GRes_1, GRes_2, GRes_3\}$. *If* $[\mathbf{A}]^\alpha$ *and* $[\mathbf{B}]^\beta$ *are proper literals, then there is a derivation* $\Delta_2 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash_{\mathcal{EP}_{fc}} \mathcal{C}_4$.

2. *Let* $\mathcal{C}_4$ *be a proper clause and* $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$ *be clauses. For each derivation* $\Delta_1 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash^r \mathcal{C}_3 \vdash_{\mathcal{CNF}} \mathcal{C}_4$ *where* $r \in \{GRes_1, GRes_2, GRes_3\}$, *there is a derivation* $\Delta_2 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash_{\mathcal{EP}_{fc}} \mathcal{C}_4$.

Proof:
(1) The proof is by induction on the length of the $\mathcal{CNF}$-derivation. In the base case the clauses $\mathcal{C}_1$ and $\mathcal{C}_2$ must be proper and thus the application of the generalised resolution rule can be replaced by an application of the ordinary resolution rule *Res* and subsequent eager unification. In the step case we simply switch the generalised resolution step with the first clause normalisation step (which is obviously applicable either to $\mathcal{C}_1$ or $\mathcal{C}_2$ as well and does operate on $[\mathbf{A}]^T$ or $[\mathbf{B}]^T$ as these literals are assumed to be proper). The assertion then follows by induction hypotheses.
(2) The proof is by induction on the number of logical connectives in the premise clauses and we discuss all three rules separately:
$\underline{GRes_1}$: In this case the clauses $\{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3\}$ are of form $\mathcal{C}_1 : [\mathbf{A}]^\alpha \vee C$, $\mathcal{C}_2 : [\mathbf{A}]^\beta \vee D$, $\mathcal{C}_3 : C \vee D$. In the base case we have that literals $[\mathbf{A}]^\alpha$ and $[\mathbf{A}]^\beta$ are proper and thus the assertion follows by (1). In the induction step we have to consider the following cases: (i) $\mathbf{A} \equiv (\neg\ \mathbf{M})$, (ii) $\mathbf{A} \equiv (\Pi\ \mathbf{M})$, and (iii) $\mathbf{A} \equiv (\vee\ \mathbf{M}\ \mathbf{N})$. Concerning the polarities we only consider the case where $\alpha \equiv T$ and $\beta \equiv F$. The argumentation for the dual constellation is analogous. Let us discuss the complicate case (iii) first. The clauses $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$ are of form $\mathcal{C}_1 : [\vee\ \mathbf{M}\ \mathbf{N}]^T \vee C$, $\mathcal{C}_2 : [(\vee\ \mathbf{M}\ \mathbf{N}]^F \vee D$, $\mathcal{C}_3 : C \vee D$. Obviously we can perform the following clause normalisation steps:

$$\frac{\mathcal{C}_1 : [\vee\ \mathbf{M}\ \mathbf{N}]^T \vee C}{\mathcal{C}_5 : [\mathbf{M}]^T \vee [\mathbf{N}]^T \vee C}\ \vee^T \qquad \text{and} \qquad \frac{\mathcal{C}_2 : [\vee\ \mathbf{M}\ \mathbf{N}]^F \vee D}{\begin{array}{l}\mathcal{C}_6 : [\mathbf{M}]^F \vee D \\ \mathcal{C}_7 : [\mathbf{N}]^F \vee D\end{array}}\ \vee^F_l, \vee^F_r$$

Thus, we can derive $\mathcal{C}_3$ (and consequently $\mathcal{C}_4$) also from clauses $\{\mathcal{C}_5, \mathcal{C}_6, \mathcal{C}_7\}$ with $GRes_1$:

$$\dfrac{\dfrac{\dfrac{\mathcal{C}_1 : [\vee\ \mathbf{M}\ \mathbf{N}]^T \vee C}{\mathcal{C}_5 : [\mathbf{M}]^T \vee [\mathbf{N}]^T \vee C}\ \vee^T \quad \dfrac{\mathcal{C}_2 : [\vee\ \mathbf{M}\ \mathbf{N}]^F \vee D}{\mathcal{C}_6 : [\mathbf{M}]^F \vee D}\ \vee^F_l}{\mathcal{C}'_3 : [\mathbf{N}]^T \vee C \vee D}\ GRes_1, Fac, Triv \quad \dfrac{\mathcal{C}_2 : [\vee\ \mathbf{M}\ \mathbf{N}]^F \vee D}{\mathcal{C}_7 : [\mathbf{N}]^F \vee D}\ \vee^F_r}{\begin{array}{c}\mathcal{C}_3 : C \vee D \\ \vdots\ \mathcal{CNF} \\ \mathcal{C}_4\end{array}}\ GRes_1$$

By induction hypotheses applied to the latter application of $GRes_1$ we know that there exists a derivation

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{C}_1}{\mathcal{C}_5}\vee^T\ \dfrac{\mathcal{C}_2}{\mathcal{C}_6}\vee_l^F}{\mathcal{C}_3'}GRes_1\quad\dfrac{\mathcal{C}_2}{\mathcal{C}_7}\vee_r^F}{\vdots\ \mathcal{EP}_{fc}}}{\mathcal{C}_4}$$

Our aim is to apply the induction hypotheses also to the first application of $GRes_1$ but unfortunately the preconditions are not met as $\mathcal{C}_3'$ is not necessarily a proper clause. Therefore we first apply lemma 5.11 which gives us that

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{C}_1}{\mathcal{C}_5}\vee^T\ \dfrac{\mathcal{C}_2}{\mathcal{C}_6}\vee_l^F}{\mathcal{C}_3'}GRes_1}{\vdots\ \mathcal{CNF}}}{\mathcal{D}_1}\quad\ldots\quad\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{C}_1}{\mathcal{C}_5}\vee^T\ \dfrac{\mathcal{C}_2}{\mathcal{C}_6}\vee_l^F}{\mathcal{C}_3'}GRes_1}{\vdots\ \mathcal{CNF}}}{\mathcal{D}_n}\quad\dfrac{\mathcal{C}_2}{\mathcal{C}_7}\vee_r^F}{\vdots\ \mathcal{EP}_{fc}}}{\mathcal{C}_4}$$

where $\mathcal{CNF}(\mathcal{C}_3') \equiv \{D_1, \ldots, D_n\}$. The preconditions for the induction hypotheses are now met for $\mathcal{D}_1, \ldots, \mathcal{D}_n$ and by applying induction hypotheses for n times we get:

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{C}_1}{\mathcal{C}_5}\vee^T\ \dfrac{\mathcal{C}_2}{\mathcal{C}_6}\vee_l^F}{\vdots\ \mathcal{EP}_{fc}}}{\mathcal{D}_1}\quad\ldots\quad\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{C}_1}{\mathcal{C}_5}\vee^T\ \dfrac{\mathcal{C}_2}{\mathcal{C}_6}\vee_l^F}{\vdots\ \mathcal{EP}_{fc}}}{\mathcal{D}_n}\quad\dfrac{\mathcal{C}_2}{\mathcal{C}_7}\vee_r^F}{\vdots\ \mathcal{EP}_{fc}}}{\mathcal{C}_4}$$

The cases (i) and (ii) are proven along the same ideas. But these cases are much simpler as we need to apply the induction hypotheses only once (cf. the cases discussed for $GRes_2$ and $GRes_3$ below) and we can omit the application of lemma 5.11.

$\underline{GRes_2}$: In this case the clauses $\{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3\}$ are of form $\mathcal{C}_1 : [\mathbf{A}\ \overline{Y^n}]^\alpha \vee C$, $\mathcal{C}_2 : [X\ \overline{T^n}]^\beta \vee D$, $\mathcal{C}_3 : (C \vee D)_{[\mathbf{A}/X, \overline{T^n}/\overline{Y^n}]}$. The argumentation is analogous to the above case for $GRes_1$ and the base case follows by (1). In the induction step we have to consider the following cases: (i) $\mathbf{A} \equiv (\neg\ Y)$, (ii) $\mathbf{A} \equiv (\Pi\ \mathbf{Y})$, (iii) $\mathbf{A} \equiv (\vee\ Y^1\ Y^2)$ and (iv) $\mathbf{A} \equiv ((\vee\ \mathbf{M})\ Y)$. As a representative we consider here the induction step for the case where $[\mathbf{A}\ \overline{Y^n}]^\alpha$ is of form $[(\neg\ Y)]^F$ and thus $[X\ \overline{T^n}]^\beta$ is of form $[X\ \mathbf{T}_o]^T$. Obviously we have that

$$\dfrac{\dfrac{\dfrac{\mathcal{C}_1 : [\neg Y]^F \vee C}{\mathcal{C}_5 : [Y]^T \vee C}\neg^F\quad\dfrac{\dfrac{\mathcal{C}_2 : [X\ \mathbf{T}]^T \vee D}{\mathcal{C}_6 : [X'\ \mathbf{T}]^F \vee D}\ Prim, Subst\qquad\mathcal{C}_2 : [X\ \mathbf{T}]^T \vee D}{\mathcal{C}_7 : [\mathbf{T}]^F \vee D}\ Res, Fac, Uni, Subst}{\mathcal{C}_3 : (C \vee D)_{[\mathbf{T}/Y]}}\ GRes_2}{\vdots\ \mathcal{CNF}}}{\mathcal{C}_4}$$

Induction hypotheses is applicable and we get that

$$\dfrac{\dfrac{\dfrac{\mathcal{C}_1 : [\neg Y]^F \vee C}{\mathcal{C}_5 : [Y]^T \vee C}\neg^F\quad\dfrac{\dfrac{\mathcal{C}_2 : [X\ \mathbf{T}]^T \vee D}{\mathcal{C}_6 : [X'\ \mathbf{T}]^F \vee D}\ Prim, Subst\qquad\mathcal{C}_2 : [X\ \mathbf{T}]^T \vee D}{\mathcal{C}_7 : [\mathbf{T}]^F \vee D}\ Res, Fac, Uni, Subst}{\vdots\ \mathcal{EP}_{fc}}}{\mathcal{C}_4}$$

which proves the assertion.

GRes$_3$: In this case the clauses $\{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3\}$ are of form $\mathcal{C}_1 : [\mathbf{A}_{\overline{\gamma}\to o}\ \overline{\mathbf{T}^n_\gamma}]^\alpha \vee C$, $\mathcal{C}_2 : [\mathbf{A}_{\overline{\gamma}\to o}\ \overline{X^n_\gamma}]^\beta \vee \mathbf{D}$, $\mathcal{C}_3 : (\mathbf{C} \vee \mathbf{D})_{[\overline{T^n}/\overline{X^n}]}$ and the base case again follows by (1). But this time we have to consider more cases in the induction step, as $[\mathbf{A}_{\overline{\gamma}\to o}\ \overline{\mathbf{T}^n_\gamma}]^\alpha$ can be of form $[\vee\ \mathbf{T}_1\ \mathbf{T}_2]^T, [\vee\ \mathbf{T}_1\ \mathbf{T}_2]^F$ where $n \equiv 2$, or $[\neg\ \mathbf{T}_1]^T, [\neg\ \mathbf{T}_1]^F, [\Pi\ \mathbf{T}_1]^T, [\Pi\ \mathbf{T}_1]^F, [(\vee\ \mathbf{A}_1)\ \mathbf{T}_1]^T, [(\vee\ \mathbf{A}_1)\ \mathbf{T}_1]^F$ where $n \equiv 1$, and finally $[\neg\ \mathbf{A}_1]^T, [\neg\ \mathbf{A}_1]^F, [\Pi\ \mathbf{A}_1]^T, [\Pi\ \mathbf{A}_1]^F, [\vee\ \mathbf{A}_1\ \mathbf{A}_2]^T, [\vee\ \mathbf{A}_1\ \mathbf{A}_2]^F$ where $n \equiv 0$. As a representative we discuss the case where $[\mathbf{A}_{\overline{\gamma}\to o}\ \overline{\mathbf{T}^n_\gamma}]^\alpha$ is of form $[\Pi\ \mathbf{T}_1]^T$ for $n \equiv 1$. Obviously we have that

$$\dfrac{\dfrac{\dfrac{\mathcal{C}_1 : [\Pi\ \mathbf{T}_1]^T \vee C}{\mathcal{C}_5 : [\mathbf{T}_1\ Y]^T \vee C}\ \Pi^T \quad \dfrac{\mathcal{C}_2 : [\Pi\ X_1]^F \vee D}{\mathcal{C}_6 : [X_1\ s]^F \vee D}\ \Pi^F}{\mathcal{C}_3 : (C \vee D)_{[\mathbf{T}_1/X_1, s/Y]}}\ GRes_2}{\begin{array}{c}\vdots\ \mathcal{CNF}\\ \mathcal{C}_4\end{array}}$$

Where Y is an new variable and s a new Skolem term of appropriate type. Induction hypotheses is applicable and we get that

$$\dfrac{\dfrac{\mathcal{C}_1 : [\Pi\ \mathbf{T}_1]^T \vee C}{\mathcal{C}_5 : [\mathbf{T}_1\ Y]^T \vee C}\ \neg^F \quad \dfrac{\mathcal{C}_2 : [\Pi\ \mathbf{T}_1]^F \vee D}{\mathcal{C}_6 : [\mathbf{T}_1\ s]^F \vee D}\ \Pi^F}{\begin{array}{c}\vdots\ \mathcal{EP}_{fc}\\ \mathcal{C}_4\end{array}}$$

which proves the assertion. □

The following paramodulation lemma now shows that also the generalised paramodulation rule *GPara* is weakly derivable in $\mathcal{EP}_{fc}$. The proof employs the (weakly derivable) generalised resolution rules $GRes_1$ and $GRes_3$.

Lemma 5.18 (Weak Derivability of Generalised Paramodulation Rule).

1. *Let $\mathcal{C}_1$ be a clause and $\mathcal{C}_2$ be a proper clause.*

   (a) $\mathcal{C}_1 : [\mathbf{A}]^\alpha \vee D$ and $\mathcal{C}_2 : [\mathbf{A} = \mathbf{B}]^T$ or

   (b) $\mathcal{C}_1 : [\mathbf{A}_{\overline{\gamma}\to o}\ \overline{\mathbf{T}^n_\gamma}]^\alpha \vee D$ and $\mathcal{C}_2 : [(\mathbf{A}_{\overline{\gamma}\to o}\ \overline{X^n_\gamma}) = (\mathbf{B}_{\overline{\gamma}\to o}\ \overline{X^n_\gamma})]^T$

   Then in case (1a) there exist a derivation $\Delta_1 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash_{\mathcal{EP}_{fc}} \mathcal{C}_3 : [\mathbf{B}]^\alpha \vee D$ and case (1b) a derivation $\Delta_2 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash_{\mathcal{EP}_{fc}} \mathcal{C}_3 : [\mathbf{B}_{\overline{\gamma}\to o}\ \overline{\mathbf{T}^n_\gamma}]^\alpha \vee D$.

2. *Let $\mathcal{C}_1 : [\mathbf{T}[\mathbf{A}]_p]^\alpha \vee D$, $\mathcal{C}_2 : [\mathbf{A} = \mathbf{B}]^T$ and $\mathcal{C}_3 : [\mathbf{T}[\mathbf{B}]_p]^\alpha \vee D$ such that there is a derivation $\Delta_1 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash^{GPara} \mathcal{C}_3 \vdash_{\mathcal{CNF}} \mathcal{C}_4$ for a proper clause $\mathcal{C}_4$ derived by clause normalisation from $\mathcal{C}_3$. Then there exists a derivation $\Delta_2 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash_{\mathcal{EP}_{fc}} \mathcal{C}_4$.*

Proof: (1) We first concentrate on case (1a) and consider the following $\mathcal{CNF}$-derivation:

$$\dfrac{\dfrac{\mathcal{C}_2 : [\mathbf{A} = \mathbf{B}]^T}{\mathcal{C}_4 : [\mathbf{A}] \Leftrightarrow [\mathbf{B}]^T}\ Equiv'}{\begin{array}{c}\vdots\ \mathcal{CNF}\\ \mathcal{C}_5 : [\mathbf{A}]^F \vee [\mathbf{B}]^T\\ \mathcal{C}_6 : [\mathbf{A}]^T \vee [\mathbf{B}]^F\end{array}}\ .$$

Depending on the polarity α of literal $[\mathbf{A}]^\alpha$ in clause $\mathcal{C}_1$ we now either apply the generalised resolution rule $GRes_1$ to $\mathcal{C}_1$ and $\mathcal{C}_5$ or to $\mathcal{C}_1$ and $\mathcal{C}_6$ thereby deriving clause $\mathcal{C}_3 : [\mathbf{B}]^\alpha \vee D$. Together

with lemma 5.17(2) we finally get that there is a derivation $\Delta_1 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash_{\mathcal{SP}_{fc}} \mathcal{C}_3$. The case (1b) is analogous as

$$\dfrac{\begin{array}{c}\mathcal{C}_2 : [(\mathbf{A}_{\overline{\gamma}\to o}\ \overline{X^n_\gamma}) = (\mathbf{B}_{\overline{\gamma}\to o}\ \overline{X^n_\gamma})]^T \\ \hline \mathcal{C}_2 : [(\mathbf{A}_{\overline{\gamma}\to o}\ \overline{X^n_\gamma}) \Leftrightarrow (\mathbf{B}_{\overline{\gamma}\to o}\ \overline{X^n_\gamma})]^T \end{array}\ \mathit{Equiv'}}{\begin{array}{c}\vdots\ \mathcal{CNF} \\ \mathcal{C}_2 : [\mathbf{A}_{\overline{\gamma}\to o}\ \overline{X^n_\gamma}]^F \vee [\mathbf{B}_{\overline{\gamma}\to o}\ \overline{X^n_\gamma})]^T \\ \mathcal{C}_2 : [\mathbf{A}_{\overline{\gamma}\to o}\ \overline{X^n_\gamma}]^T \vee [\mathbf{B}_{\overline{\gamma}\to o}\ \overline{X^n_\gamma})]^f \end{array}}$$

and thus the application of $GRes_3$ to $\mathcal{C}_1$ and $\mathcal{C}_5$ or to $\mathcal{C}_1$ and $\mathcal{C}_6$ derives clause $\mathcal{C}_3 : [\mathbf{B}_{\overline{\gamma}\to o}\ \overline{\mathbf{T}^n_\gamma}]^\alpha \vee D_1$. Again the assertion follows by lemma 5.17(2).

(2) The tedious proof is by induction on the length n of the embedded $\mathcal{CNF}$ derivation $\mathcal{C}_3 \vdash_{\mathcal{CNF}} \mathcal{C}_4$.

$\underline{n = 0}$: In the base case $\mathcal{C}_3$ must be a proper clause, i.e. the clause rests D consist only of proper literals and the literal $[\mathbf{T}[\mathbf{B}]_p]^\alpha$ is proper. We now consider the possible cases for literal $[\mathbf{T}[\mathbf{A}]]^\alpha$.

1. $[\mathbf{T}[\mathbf{A}]_p]^T$ is a proper literal and $\mathcal{C}_1$ is a proper clause. Instead of general paramodulation we can apply in this case the standard paramodulation rule *Para* to $\mathcal{C}_1$ in order to derive $\mathcal{C}_4$ by $\Delta_2 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash^{Para} [\mathbf{T}[\mathbf{B}]_p]^T \vee D_1 \vee [\mathbf{A} = \mathbf{A}]^F \vdash^{Triv} \mathcal{C}_4$.

2. $[\mathbf{T}[\mathbf{A}]_p]^F$ is a proper literal and $\mathcal{C}_1$ is a proper clause. If $[\mathbf{T}[\mathbf{A}]_p]^F$ is not a unification constraint then the argumentation is analogous to the above case. In case $[\mathbf{T}[\mathbf{A}]_p]^F$ is a unification constraint it is (without loss of generality) of form $[T_1[\mathbf{A}] = T_2]^F$. In this case we employ a derivation that is analogous to the one already discussed in remark 5.2:

$$\begin{array}{ll} \mathit{Leib}(\mathcal{C}_1), \mathcal{CNF} : & \mathcal{D}_1 : C \vee [p\ \mathbf{T}_1[\mathbf{A}_\alpha]]^T \\ & \mathcal{D}_2 : C \vee [p\ \mathbf{T}_2]^F \\ \mathit{Para}(\mathcal{D}_1, \mathcal{C}_2), \mathit{Triv} : & \mathcal{D}_3 : C \vee [p\ \mathbf{T}_1[\mathbf{B}_\alpha]]^T \\ \mathit{Res}(\mathcal{D}_3, \mathcal{D}_2), \mathit{Fac}, \mathit{Triv} : & \mathcal{D}_4 : C \vee D \vee [(p\ \mathbf{T}_1[\mathbf{B}_\alpha]) = (p\ \mathbf{T}_2)]^F \\ \mathit{Dec}(\mathcal{D}_4), \mathit{Triv} : & \mathcal{D}_5 : C \vee D \vee [\mathbf{T}_1[\mathbf{B}_\alpha] = \mathbf{T}_2]^F \end{array}$$

3. $[\mathbf{T}[\mathbf{A}]_p]^\alpha$ is not a proper literal, i.e. has a logical connective different from $=$ at head position. We consider the possible subterm positions p of term $\mathbf{A}$ in $\mathbf{T}$.

   (a) As this would contradict our assumption ($[\mathbf{T}[\mathbf{B}]_p]^\alpha$ is a proper literal) we know that for literal $[\mathbf{T}[\mathbf{A}]_p]^\alpha$ the following cases are impossible: $[\neg\ (\mathbf{M}[\mathbf{A}]_{p'})]^\alpha$, $[\vee\ (\mathbf{M}[\mathbf{A}]_{p'})\ \mathbf{N}]^\alpha$, $[\vee\ \mathbf{M}\ (\mathbf{N}[\mathbf{A}]_{p'})]^\alpha$ and $[\Pi\ (\mathbf{M}[\mathbf{A}]_{p'})]^\alpha$.

   (b) Literal $[\mathbf{T}[\mathbf{A}]_p]^\alpha$ has form $[\overbrace{\neg\ \mathbf{M}}^{\mathbf{A}}]^\alpha$, $[\overbrace{\vee\ \mathbf{M}\ \mathbf{N}}^{\mathbf{A}}]^\alpha$ or $[\overbrace{\Pi\ \mathbf{M}}^{\mathbf{A}}]^\alpha$. Then $\mathcal{C}_2$ is of form $[(\neg\mathbf{M}) = \mathbf{B}]^\alpha$, $[(\vee\ \mathbf{M}\ \mathbf{N}) = \mathbf{B}]^\alpha$ and $[(\Pi\ \mathbf{M}) = \mathbf{B}]^\alpha$ respectively. For all these cases the assertion now follows by (1).

   (c) Literal $[\mathbf{T}[\mathbf{A}]_p]^\alpha$ has form $[\overbrace{\neg}^{\mathbf{A}}\ \mathbf{M}]^\alpha$, $[\overbrace{\vee}^{\mathbf{A}}\ \mathbf{M}\ \mathbf{N}]^\alpha$, $[\overbrace{\Pi}^{\mathbf{A}}\ \mathbf{M}]^\alpha$ or $[\overbrace{(\vee\ \mathbf{M})}^{\mathbf{A}}\ \mathbf{N}]^\alpha$. Then analogous to above $\mathcal{C}_2$ is of form $[\neg = \mathbf{B}]^\alpha$, $[\vee = \mathbf{B}]^\alpha$, $[\Pi = \mathbf{B}]^\alpha$ and $[(\vee\ \mathbf{M}) = \mathbf{B}]^\alpha$ respectively. Applying positive extensionality rule *Func'* to these clauses leads to $[(\neg\ X) = (\mathbf{B}\ X)]^\alpha$, $[(\vee\ X\ Y) = (\mathbf{B}\ X\ Y)]^\alpha$, $[(\Pi\ P) = (\mathbf{B}\ P)]^\alpha$ and $[(\vee\ \mathbf{M}\ X) = (\mathbf{B}\ X)]^\alpha$. For all these cases the assertion now follows by (1).

$\underline{n > 0}$: In the induction step we know that literal $[\mathbf{T}[\mathbf{B}]_p]^\alpha$ must have a logical connective at head position, as otherwise the length of the $\mathcal{CNF}$-derivation is 0. We now distinguish the following two cases:

1. At least one of the literals in D is not proper. In this case it is obvious that we can reorder the $\mathcal{CNF}$-derivation such that one of the clause normalisation steps that modifies D comes first. Without loss of generality let us assume that $\mathcal{CNF}$-rule r transforms in step

k $(0 < k < n)$ of derivation Δ_1 the clause rest D into D'. Then we can reorder derivation Δ_1 and obtain a derivation $\Delta_3 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash^{GPara} \mathcal{C}_3 \vdash^r \mathcal{C}_3' : [\mathbf{T}[\mathbf{B}]_p]^\alpha \vee D' \vdash_{\mathcal{CNF}} \mathcal{C}_4$ where rule r is applied first. Obviously we can also switch the first two steps in Δ_3 such that we get $\Delta_4 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash^r \{\mathcal{C}_1' : [\mathbf{T}[\mathbf{A}]_p]^\alpha \vee D', \mathcal{C}_2\} \vdash^{GPara} \mathcal{C}_3' \vdash_{\mathcal{CNF}} \mathcal{C}_4$. Now induction hypotheses is applicable and we get that there is $\mathcal{EP}_{fc}$-derivation $\Delta_5 : \{\mathcal{C}_1', \mathcal{C}_2\} \vdash_{\mathcal{EP}_{fc}} \mathcal{C}_4$.

2. D consist only of proper literals. We again examine the structure of literal $[\mathbf{T}[\mathbf{A}]_p]^\alpha$ and distinguish between all possible subterm positions p of term $\mathbf{A}$ in $\mathbf{T}$.

   (a) Literal $[\mathbf{T}[\mathbf{A}]_p]^\alpha$ is of form $[\neg\ (\mathbf{M}[\mathbf{A}]_{p'})]^\alpha$, $[\vee\ (\mathbf{M}[\mathbf{A}]_{p'})\ \mathbf{N}]^\alpha$, $[\vee\ \mathbf{M}\ (\mathbf{N}[\mathbf{A}]_{p'})]^\alpha$ or $[\Pi\ (\mathbf{M}[\mathbf{A}]_{p'})]^\alpha$. As the proofs are analogous we only present the following case here. Assume $\mathcal{C}_1$ is of form $[\vee\ \mathbf{M}\ (\mathbf{N}[\mathbf{A}]_{p'})]^T \vee D$. Consider now derivation Δ_1 which obviously applies the $\mathcal{CNF}$-rule $\vee^T$ first within the normalisation process (note that D contains only proper literals). Thus we have $\Delta_1 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash^{GPara} \mathcal{C}_3 \vdash^{\vee^T} \{\mathcal{C}_3' : [\mathbf{M}]^T \vee [\mathbf{N}[\mathbf{B}]_p]^T \vee D \vdash_{\mathcal{CNF}} \mathcal{C}_4$. We can obviously switch the first two steps in Δ_1 such that we get an alternative derivation $\Delta_1' : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash^{\vee^T} \{\mathcal{C}_1' : [\mathbf{M}]^T \vee [\mathbf{N}[\mathbf{A}]_p]^T \vee D, \mathcal{C}_2\} \vdash^{GPara} \mathcal{C}_3' \vdash_{\mathcal{CNF}} \mathcal{C}_4$. By induction hypotheses we now get that there is a derivation $\Delta_2' : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash^{\vee^T} \{\mathcal{C}_1', \mathcal{C}_2\} \vdash_{\mathcal{EP}_{fc}} \mathcal{C}_4$.

   (b) Literal $[\mathbf{T}[\mathbf{A}]_p]^\alpha$ is of form $[\overbrace{\neg\ \mathbf{M}}^{\mathbf{A}}]^\alpha$, $[\overbrace{\vee\ \mathbf{M}\ \mathbf{N}}^{\mathbf{A}}]^\alpha$ or $[\overbrace{\Pi\ \mathbf{M}}^{\mathbf{A}}]^\alpha$. Then $\mathcal{C}_2$ is of form $[(\neg\ \mathbf{M}) = \mathbf{B}]^\alpha$, $[(\vee\ \mathbf{M}\ \mathbf{N}) = \mathbf{B}]^\alpha$ or $[(\Pi\ \mathbf{M}) = \mathbf{B}]^\alpha$. In all cases the assertion follows immediately by (1).

   (c) Literal $[\mathbf{T}[\mathbf{A}]_p]^\alpha$ is of form $[\overbrace{\neg}^{\mathbf{A}}\ \mathbf{M}]^\alpha$, $[\overbrace{\vee}^{\mathbf{A}}\ \mathbf{M}\ \mathbf{N}]^\alpha$, $[\overbrace{\Pi}^{\mathbf{A}}\ \mathbf{M}]^\alpha$ or $[\overbrace{(\vee\ \mathbf{M})}^{\mathbf{A}}\ \mathbf{N}]^\alpha$ Then analogous to above $\mathcal{C}_2$ is of form $[\neg = \mathbf{B}]^\alpha$, $[\vee = \mathbf{B}]^\alpha$, $[\Pi = \mathbf{B}]^\alpha$ and $[(\vee\ \mathbf{M}) = \mathbf{B}]^\alpha$ respectively. Applying positive extensionality rule *Func'* to these clauses leads to $[(\neg\ X) = (\mathbf{B}\ X)]^\alpha$, $[(\vee\ X\ Y) = (\mathbf{B}\ X\ Y)]^\alpha$, $[(\Pi\ P) = (\mathbf{B}\ P)]^\alpha$ and $[(\vee\ \mathbf{M}\ X) = (\mathbf{B}\ X)]^\alpha$. For all these cases the assertion now follows by (1).

□

As □ is just a special proper clause we immediately get the following corollary.

Corollary 5.19 (Admissibility of Generalised Paramodulation Rule). *The generalised paramodulation rule* GPara *is admissible in calculus* $\mathcal{EP}_{fc}$.

Theorem 5.20 (Completeness of $\mathcal{EP}_{fc}$). *The calculi* $\mathcal{EP}_{fc}$ *is complete with respect to Henkin models.*

Proof:

Let Γ_Σ be the set of Σ-sentences which cannot be refuted by the calculus $\mathcal{EP}_{fc}$ ($\Gamma_\Sigma := \{\Phi \subseteq cwff_o(\Sigma) | \Phi_{cl} \not\vdash_{\mathcal{EP}_{fc}} \square\}$), then we show that Γ_Σ is a saturated abstract consistency class for Henkin models with primitive equality (c.f. 2.3). This entails completeness with the model existence theorem for Henkin models with primitive equality 2.1(2).

First we have to verify that Γ_Σ validates the abstract consistency properties ∇_c, $\nabla_\neg$, ∇_β, $\nabla_\vee$, $\nabla_\wedge$, $\nabla_\forall$, $\nabla_\exists$, $\nabla_\mathfrak{b}$, $\nabla_\mathfrak{q}$ and that Γ_Σ is saturated. For all of these cases the proofs are identical to the corresponding argumentations in theorem 3.16. The only difference is that we employ the lemmata 5.14(1)-(2) and 5.12 instead of 3.15(1)-(2) and 3.12. Thus, all we need to ensure is the validity of the additional abstract consistency property $\nabla_\mathfrak{e}$ for primitive equality.

$\nabla_\mathfrak{e}$ (r) $\neg(\mathbf{A} =^\alpha \mathbf{A}) \notin \Phi$
(s) if $\mathbf{F}[\mathbf{A}]_p \in \Phi$ and $\mathbf{A} = \mathbf{B} \in \Phi$, then $\Phi * \mathbf{F}[\mathbf{B}]_p \in \Gamma_\Sigma$

(r) We have that $[\mathbf{A} =^\alpha \mathbf{A}]^F \vdash^{Triv} \square$ and thus $\neg(\mathbf{A} =^\alpha \mathbf{A})$ cannot be in Φ.

(s) Analogous to the cases in 3.16 we show the contrapositive of the assertion and thus we assume that there is derivation $\Delta_1 : \Phi_{cl} * [\mathbf{F}[\mathbf{B}]_p]^T \vdash_{\mathcal{EP}_{fc}} \square$ Now consider the following $\mathcal{EP}_{fc}$-derivation:

$$\frac{[\mathbf{F}[\mathbf{A}]_p]^T \quad [\mathbf{A} = \mathbf{B}]^T}{[\mathbf{F}[\mathbf{B}]_p]^T}\ GPara$$

By corollary 5.19 we know that the generalised paramodulation rule *GPara* is admissible for calculus $\mathcal{EP}_{fc}$ and thus there is a $\mathcal{EP}_{fc}$-derivation $\{[\mathbf{F}[\mathbf{B}]_p]^T, [\mathbf{A} = \mathbf{B}]^T\} \vdash_{\mathcal{EP}_{fc}} [\mathbf{F}[\mathbf{B}]_p]^T$. Consequently there is a derivation $\Delta_1 : \Phi * [\mathbf{F}[\mathbf{A}]_p]^T * [\mathbf{A} = \mathbf{B}]^T \vdash_{\mathcal{EP}_{fc}} \Phi_{cl} * [\mathbf{F}[\mathbf{A}]_p]^T * [\mathbf{A} = \mathbf{B}]^T * [\mathbf{F}[\mathbf{B}]_p]^T \vdash_{\mathcal{EP}_{fc}} \square$ which completes the proof. □

5.6 Theorem Equivalence

Analogously to subsection 3.5 we now prove that $\mathcal{EP}_{fc}$ and $\mathcal{EP}_f$ are theorem equivalent. Theorem equivalence of $\mathcal{EP}_{fc}$ and $\mathcal{EP}$ is then presented as a conjecture that has not yet been formally proven yet.

Lemma 5.21 (Proper Clauses in $\mathcal{EP}$ and $\mathcal{EP}_f$). *For each non-proper clause $\mathcal{C}$ and clause set Φ such that $\Phi * \vdash_{\mathcal{EP}_{fc}} \mathcal{C}$ we have that $\Phi \vdash_{\mathcal{EP}_f} \mathcal{C}$.*

Proof: Analogous to lemma 3.20. In the induction step the case for rule *Para* is treated analogously to the cases $r \in \{Res, Fac, Prim\}$ and the rules *Func'* and *Equiv'* are treated analogously to the unification rules. □

As □ is also a proper clause we immediately get the following corollary:

Corollary 5.22 (Theorem Equivalence of $\mathcal{EP}_{fc}$ and $\mathcal{EP}_f$). *The calculi $\mathcal{EP}_{fc}$ and $\mathcal{EP}_f$ are theorem equivalent.*

Conjecture 5.23 (Theorem Equivalence of $\mathcal{EP}$ and $\mathcal{EP}_{fc}$ (or $\mathcal{EP}_f$)). *The calculi $\mathcal{EP}$ and $\mathcal{EP}_{fc}$ (or $\mathcal{EP}_f$) are theorem equivalent.*

Obviously the proof for the latter conjecture will be analogous to the proof of the theorem equivalence for calculi $\mathcal{ER}_{fc}$ and $\mathcal{ER}$ and thus also gains evidence by the examples already carried out with the LEO-prover [BK98b] for extensional higher-order resolution as well as the new challenging examples for extensional higher-order paramodulation discussed in section 7.

6 Extensional Higher-Order RUE-Resolution: $\mathcal{ERUE}$

6.1 Resolution on Unification Constraints

In this section we will adapt the **R**esolution by **U**unification and **E**quality approach [Dig79] to our higher-order setting. The key idea is to allow resolution and factorisation rules also to operate on unification constraints. This implements the main ideas of first-order RUE-resolution directly in our higher-order calculus. More precisely our approach allows to compute partial E-unifiers with respect to a specified theory E (where E is specified in form of a set of unitary or even conditional equations in clause form in the search space) by employing resolution on unification within constraint the calculus itself. This is due to the fact that the extensional higher-order resolution approach already realizes a test calculus for general higher-order E-pre-unification (or general higher-order E-unification in case we also add rule *FlexFlex*). Furthermore, each partial E-(pre-)unifier can be applied to a clause with rule *Subst*. Finally, like in the traditional first-order RUE-resolution approach, the non-solved unification constraints are coded as still open unification constraints within the particular clauses.

*Remark 6.1 (**Resolution on Unification Constraints**).* The idea to resolve on unification constraints arised while discussing the necessity of Rule *Leib* within extensional higher-order resolution with Frank Pfenning during a stay at Carnegie Mellon University in 1997. At that time the

author not interested in a calculus for primitive equality but focused on the question whether it is possible to avoid rule *Leib* in calculus $\mathcal{ER}$. And as is demonstrated by an example in [Ben97] at least some applications of rule *Leib* become superfluous in $\mathcal{ER}$ if one allows resolution and factorisation on unification constraints. Thus, it seems to be a very interesting task to examine whether rule *Leib* is admissible in the extensional higher-order RUE-resolution approach $\mathcal{ERUE}$. Because lack of time this question has not been investigated in detail yet. Anyway, the idea of resolution on unification constraints turned out to be a very fruitful idea.

Remark 6.2 ***(Incompleteness of a naive RUE-Resolution approach).*** Analogous to our a naive adaption of the first-order paramodulation approach, we obtain a Henkin incomplete RUE-resolution approach if we not additionally add the positive extensionality to our calculus. This is nicely illustrated by example $\mathbf{E}_1^{Para}$ already used in the incompleteness proof 5.5 for $\mathcal{EP}$. The reader may easily check that no single rule is applicable to the unit clause $\mathcal{C}_1 : [a = \neg a]^T$.

6.2 Basic Definitions

Analogously to section 3 we first further extend our extensional higher-order RUE-resolution approach by adding rule *FlexFlex* and lifting the single clause normalisation rules to calculus level instead of grouping exhaustive clause normalisation derivations together with rule *Cnf*.

The definitions for clause normalisation calculus $\mathcal{CNF}$ (see 3.2), $\mathcal{UNI}$ and $\mathcal{UNI}_f$ (see 3.4) need not to be modified and the lemmata 3.3 (soundness of $\mathcal{CNF}$) and 3.5, 3.6 (properties of higher-order unification) will be employed in this section again.

Definition 6.3 (Extensional Higher-Order RUE-Resolution).

$\mathcal{ERUE}$ The calculus $\mathcal{ERUE}$ consists of the rules of calculus $\mathcal{ER}$ (see figure 2) enriched by the positive extensionality rules displayed in figure 5. The most important aspect is that we allow to resolve and factorise on unification constraints. Furthermore we want to point out again that unification constraints are assumed to be symmetric which could also be formulated by the following rule:

$$\frac{\mathcal{C}_1 : C \vee [\mathbf{A} = \mathbf{B}]^F}{\mathcal{C}_2 : C \vee [\mathbf{B} = \mathbf{A}]^F}\ sym$$

We employ this convention despite the fact that the symmetry rule *sym* is derivable (c.f. remark 6.4) in $\mathcal{ERUE}$ — as well as in $\mathcal{ER}$ and $\mathcal{EP}$ — as it shortens and eases derivations in practice and is with respect of its complexity acceptable.

$\mathcal{ERUE}_f$ The extension $\mathcal{ERUE}_f$ of calculus $\mathcal{ERUE}$ that employs full higher-order unification instead of higher-order pre-unification is defined as $\mathcal{ERUE}_f := \mathcal{ERUE} \cup \{FlexFlex\}$.

$\mathcal{ERUE}_{fc}$ The calculus $\mathcal{ERUE}_{fc}$ that employs stepwise instead of exhaustive clause normalisation is defined by $\mathcal{ERUE}_{fc} := (\mathcal{ERUE}_f \backslash \{Cnf\}) \cup \mathcal{CNF}$.

We furthermore assume that the result of each rule application is transformed into head-normal form. A set of formulas Φ is refutable in calculus $\mathcal{ERUE}$ iff there is a derivation $\Delta : \Phi_{cl} \vdash_{\mathcal{ERUE}} \square$ where $\Phi_{cl} := \{[\mathbf{F}]^T | \mathbf{F} \in \Phi\}$ is the set obtained from Φ by simple pre-clausification.

Remark 6.4 ***(Derivability of symmetry rule).*** The following derivation shows that the symmetry rule is derivable in $\mathcal{ERUE}$ ($\mathcal{ER}$ and $\mathcal{EP}$):

$$\begin{array}{ll} Leib(\mathcal{C}_1) : & \mathcal{D}_1 : C \vee [p\ \mathbf{A}]^T \\ & \mathcal{D}_2 : C \vee [p\ \mathbf{B}]^F \\ Res(\mathcal{D}_2, \mathcal{D}_1), Fac, Triv : & \mathcal{D}_3 : C \vee [p\ \mathbf{B} = p\ \mathbf{A}]^F \\ Dec(\mathcal{D}_3), Triv : & \mathcal{C}_2 : C \vee [\mathbf{B} = \mathbf{A}]^F \end{array}$$

Theorem 6.5 (Soundness of Extensional Higher-Order RUE-Resolution). *The calculi $\mathcal{ERUE}_{fc}$, $\mathcal{ERUE}_f$ and $\mathcal{ERUE}$ are sound with respect to standard semantics.*

Proof: Soundness of most of our calculus rules have already been discussed in lemmata 3.9 and 5.10. We additionally have to verify that resolution and factorisation on unification constraints is sound. Note that unification constraints are treated as ordinary negative literals with a primitive equality symbol at head position. Thus there is nothing new to show here and we can employ the standard argumentation for soundness of the resolution and factorisation rule. □

Lemma 6.6 (Proper Derivations). *For each non-proper clause $\mathcal{C}$, proper clause $\mathcal{C}'$ and clause set Φ such that $\Phi * \mathcal{C} \vdash_{\mathcal{ERUE}_{fc}} \mathcal{C}'$ we have that $\Phi \cup \mathcal{CNF}(\mathcal{C}) \vdash_{\mathcal{EP}_{fc}} \mathcal{C}'$.*

Proof: Analogous to lemma 3.10 and 5.11. It does not cause any problems that we allow to resolve on unification constraints. □

6.3 Lifting Properties

The clause normalisation lifting lemma 3.11 is not affected by the modifications to the calculus and will be employed in the following lifting lemma for calculus $\mathcal{ERUE}_{fc}$ as well.

And the proof of the adapted main lifting lemma is analogous to the one presented for $\mathcal{ER}_{fc}$. Resolution on unification constraints does not cause a new problem as primitive substitution rule *Prim* can be employed to unblock the abstract derivation in case a proper unification constraint is given on the instantiated level and a corresponding flexible negative literal is given at the uninstantiated level causing the blocking situation.

Lemma 6.7 (Lifting Lemma for $\mathcal{ERUE}_{fc}$). *Let Φ be a set of clauses, $\mathcal{D}_1$ be a clause and σ a substitution. We have that:*

1. *For each derivation $\Delta_1 : \Phi_\sigma \vdash^1_{\mathcal{ERUE}_{fc}} \mathcal{D}_1$ there exists a substitution δ, a clause $\mathcal{D}_2$ and a derivation $\Delta_2 : \Phi \vdash_{\mathcal{ERUE}_{fc}} \mathcal{D}_2$ such that $(\mathcal{D}_2)_\delta \equiv \mathcal{D}_1$.*
2. *For each derivation $\Delta_1 : \Phi_\sigma \vdash_{\mathcal{ERUE}_{fc}} \mathcal{D}_1$ there exists a substitution δ, a clause $\mathcal{D}_2$ and a derivation $\Delta_2 : \Phi \vdash_{\mathcal{ERUE}_{fc}} \mathcal{D}_2$ such that $(\mathcal{D}_2)_\delta \equiv \mathcal{D}_1$. (Note that this claim is stronger than: Φ is refutable by $\mathcal{ERUE}_{fc}$, provided that Φ_σ is.)*

Proof:
(1) $\mathcal{ERUE}_{fc}$ only slightly modifies calculus $\mathcal{EP}_{fc}$ as it does not employ rule *Para* but a slightly extended resolution instead. As mentioned above this does not cause any problems new problems and thus the argumentation is analogous to 5.12(1). The main idea is again to employ rule *Prim* in order to introduce logical connectives at head position such that the needed structure becomes available at the abstract level as well.
(2) Analogous to 5.12(2) □

6.4 Completeness

Analogously to the subsections 3.4 and 5.5 we analyse in this subsection Henkin completeness of calculus $\mathcal{ERUE}_{fc}$. The calculi $\mathcal{ERUE}_f$ and $\mathcal{ERUE}$ are then examined in subsection 6.5. Like in the case of extensional higher-order resolution theorem equivalence between $\mathcal{ERUE}$ and $\mathcal{ERUE}_{fc}$ has not been proven formally yet and will only be presented as a conjecture.

We first adapt the two lemmata 5.14 and 5.17. The former lemma compares refutability of clause sets in head-normal form with the refutability of clauses in $\beta\eta$-normal form and the latter one discusses some important refutational properties.

Lemma 6.8 (Head-Normal Form). *Let Φ be a set of clauses. If $\Phi_{\downarrow_{\beta\eta}} \vdash_{\mathcal{ERUE}_{fc}} \square$ then $\Phi_{\downarrow_h} \vdash_{\mathcal{ERUE}_{fc}} \square$*

Proof: The proof is analogous to lemma 3.13. In the induction step we additionally have to consider the cases where $r \in \{Para, Func', Equiv'\}$ which are analogous to the cases where $r \in \{Prim\} \cup \mathcal{CNF}$. □

Lemma 6.9. *Let Φ be a set of Σ-sentences and Φ_{cl} be the corresponding set of pre-clauses. Furthermore let $\mathbf{A}, \mathbf{B}$ be formulas and $\mathcal{C}, \mathcal{D}$ be clauses. We have that:*

1. *If $\Phi_{cl} * [\mathbf{A}]^T \vdash_{\mathcal{ERUE}_{fc}} \square$ and $\Phi_{cl} * [\mathbf{B}]^T \vdash_{\mathcal{ERUE}_{fc}} \square$ then $\Phi_{cl} * [\mathbf{A} \vee \mathbf{B}]^T \vdash_{\mathcal{ERUE}_{fc}} \square$.*
2. *If $\Phi_{cl} * [\mathbf{A}]^T * [\mathbf{B}]^F \vdash_{\mathcal{ERUE}_{fc}} \square$ and $\Phi_{cl} * [\mathbf{A}]^F * [\mathbf{B}]^T \vdash_{\mathcal{ERUE}_{fc}} \square$ then $\Phi_{cl} * [\mathbf{A} \Leftrightarrow \mathbf{B}]^F$*

Proof: Analogous to lemma 3.15 and 5.14. The new or slightly modified rules do not affect the argumentations. □

Analogously to section 5 we will now show that the three generalised resolution rules $GRes_1, GRes_2$ and $GRes_3$ are in a weak sense (modulo subsequent clause normalisation) derivable in calculus $\mathcal{ERUE}_{fc}$. This fact will then be employed to establish the admissibility of generalised resolution rule *GPara* for calculus $\mathcal{ERUE}_{fc}$. We will finally use the latter result in the main completeness proof for calculus $\mathcal{ERUE}_{fc}$.

Lemma 6.10 (Weak Derivability of Generalised Resolution Rules).

1. *Let $\mathcal{C}_1 : [\mathbf{A}]^\alpha \vee \mathbf{C}$ and $\mathcal{C}_2 : [\mathbf{B}]^\beta \vee \mathbf{D}$ be clauses such that there are clauses $\mathcal{C}_3 : \mathbf{C}' \vee \mathbf{D}'$ and $\mathcal{C}_4$ and a derivation $\Delta_1 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash^r \mathcal{C}_3 \vdash_{\mathcal{CNF}} \mathcal{C}_4$. If $[\mathbf{A}]^\alpha$ and $[\mathbf{B}]^\beta$ are proper literals, then there is a derivation $\Delta_2 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash_{\mathcal{ERUE}_{fc}} \mathcal{C}_4$.*
2. *For each proper clause $\mathcal{C}_4$, (pre-)clauses $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$ and derivation $\Delta_1 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash^r \mathcal{C}_3 \vdash_{\mathcal{CNF}} \mathcal{C}_4$ where $r \in \{GRes1, GRes2, GRes3\}$, there is a derivation $\Delta_2 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash_{\mathcal{ERUE}_{fc}} \mathcal{C}_4$.*

Proof: The argumentation is analogous to 5.17 and resolution on unification constraints does not cause any serious problems. □

The following paramodulation lemma now shows that the generalised paramodulation rule *GPara* is admissible in $\mathcal{ERUE}_{fc}$.

Whereas statement 6.11(1) is analogous to lemma 5.18(1) we can prove in 6.11(2) only admissibility instead of the weak derivability property in 5.18(2). This is because in 5.18 we were able to reduce the applications of the generalised paramodulation rule either to the generalised resolution rules and thus finally to the proper resolution rule *Res* or to the proper paramodulation rule *Para*. Whereas all reductions to rule *Res* are analogous here as well, we cannot employ the reductions to the proper paramodulation *Para* here. Instead we have to show that alternative reductions are possible which employ the RUE-resolution idea to resolve against unification constraints. The latter causes the loss of the weak derivability property. But fortunately admissibility is sufficient for our purposes.

Lemma 6.11 (Admissibility of Generalised Paramodulation Rule).

1. *Let $\mathcal{C}_1$ be a clause and $\mathcal{C}_2$ be a proper clause.*

   (a) $\mathcal{C}_1 : [\mathbf{A}]^\alpha \vee D$ and $\mathcal{C}_2 : [\mathbf{A} = \mathbf{B}]^T$ or

   (b) $\mathcal{C}_1 : [\mathbf{A}_{\overline{\gamma} \to o}\ \overline{\mathbf{T}^n_\gamma}]^\alpha \vee D$ and $\mathcal{C}_2 : [(\mathbf{A}_{\overline{\gamma} \to o}\ \overline{X^n_\gamma}) = (\mathbf{B}_{\overline{\gamma} \to o}\ \overline{X^n_\gamma})]^T$

   Then in case (1a) there exist a derivation $\Delta_1 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash_{\mathcal{ERUE}_{fc}} \mathcal{C}_3 : [\mathbf{B}]^\alpha \vee D$ and case (1b) a derivation $\Delta_2 : \{\mathcal{C}_1, \mathcal{C}_2\} \vdash_{\mathcal{ERUE}_{fc}} \mathcal{C}_3 : [\mathbf{B}_{\overline{\gamma} \to o}\ \overline{\mathbf{T}^n_\gamma}]^\alpha \vee D$.
2. *Let $\mathcal{C}_1[A/B]_\mathcal{P}$ be a clause that is obtained from clause $\mathcal{C}$ be replacing the occurrences of term $\mathbf{B}$ at positions $p \in \mathcal{P}$ by term $\mathbf{A}$ and let Φ be a set of clauses. If $\Delta_1 : \Phi * \mathcal{C}_1[\mathbf{A}/\mathbf{B}]_\mathcal{P} * [\mathbf{A} = \mathbf{B}]^T \vdash^{GPara} \Phi * \mathcal{C}_1[\mathbf{A}/\mathbf{B}]_\mathcal{P} * [\mathbf{A} = \mathbf{B}]^T * \mathcal{C}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}\setminus\{p\}} \vdash_{\mathcal{ERUE}_{fc}} \square$ then there is a derivation $\Delta_2 : \Phi * [\mathbf{A} = \mathbf{B}]^T \vdash_{\mathcal{ERUE}_{fc}} \square$.*

Proof:
(1) In both cases (1a) and (1b) the proof is analogous to the corresponding argumentation in lemma 5.18(1). The only difference is that we employ lemma 6.10 instead of 5.17.
(2) The proof is by induction on the length of Δ_1. Here the base step is the most complicate case as applications of the non-generalised paramodulation rule *Para* have to be replaced by derivations employing the RUE-resolution idea.

$\underline{n \equiv 1}$: If $\square \in \Phi$ the assertion follows trivially, therefore let us assume that $\square \equiv \mathcal{C}_1[\mathbf{A/B}]_{\mathcal{P}\setminus\{p\}}$. Hence $\mathcal{C}_1[\mathbf{A/B}]_{\mathcal{P}}$ has form $\mathcal{C}_1'[\mathbf{A/B}]_{\mathcal{P}_1} \vee [\mathbf{T}[\mathbf{A/B}]_{\mathcal{P}_2}]^\alpha$ where all but one literal must be flex-flex unification constraints. Without loss of generality we assume that this is the literal $[\mathbf{T}[\mathbf{A/B}]_{\mathcal{P}_2}]^\alpha$. Then the new clause $\mathcal{C}_1[\mathbf{A/B}]_{\mathcal{P}\setminus\{p\}}$ looks like $\mathcal{C}_1'[\mathbf{A/B}]_{\mathcal{P}_1} \vee [\mathbf{T}[\mathbf{A/B}]_{\mathcal{P}_2\setminus\{p'\}}]^\alpha$ where position p' in literal $[\mathbf{T}[\mathbf{A/B}]_{\mathcal{P}_2\setminus\{p'\}}]^\alpha$ specifies the position the replacement has been taken place. Obviously this clause must consist only of flex-flex unification constraints (and hence polarity α must be F) as otherwise it would be different from $\square$. Thus, the literal $[\mathbf{T}[\mathbf{A/B}]_{\mathcal{P}_2\setminus\{p'\}}]^\alpha$ is now a flex-flex constraint. We now consider the position p' in literal $[\mathbf{T}[\mathbf{A/B}]_{\mathcal{P}_2}]^F$ where the generalised paramodulation step has been applied to. The first two possibilities concentrate on replacements that include the head position in this literal.

- If p' specifies the replacement of the whole atom of the literal in focus, i.e. $\mathbf{T}[\mathbf{A/B}]_{\mathcal{P}_2} \equiv \mathbf{A}$, then we get the assertion by (1a) which means that we replace the generalised paramodulation step by a derivation that employs the generalised resolution rules instead.
- If p' specifies the replacement of a proper prefix of the atom $\mathbf{T}[\mathbf{A/B}]_{\mathcal{P}_2}$ (i.e. $\mathbf{T}[\mathbf{A/B}]_{\mathcal{P}_2}$ looks like $(\mathbf{A}\ \overline{\mathbf{U}^n})$) then we first apply the positive functional extensionality rule *Func'* for n-times to $[\mathbf{A} = \mathbf{B}]^T$ thereby generating a proper clause $[\mathbf{A}\ \overline{\mathbf{X}^n} = \mathbf{B}\ \overline{\mathbf{X}^n}]^T$. Now (1b) is applicable which means that we again replace the generalised paramodulation step by a derivation that employs the generalised resolution rules and which gives us the assertion.

Next we examine the cases where a proper subterm of the atom of $[\mathbf{T}[\mathbf{A/B}]_{\mathcal{P}_2}]^F$ gets replaced. As the replacement must lead to a flex-flex unification constraint we already know that $[\mathbf{T}[\mathbf{A/B}]_{\mathcal{P}_2}]^F$ must be a unification constraint and thus must have form $[\mathbf{T}_1[\mathbf{A/B}]_{\mathcal{P}_3} = \mathbf{T}_2[\mathbf{A/B}]_{\mathcal{P}_4}]^F$.

- If p' refers to proper subterm of either $\mathbf{T}_1[\mathbf{A/B}]_{\mathcal{P}_3}$ or $\mathbf{T}_2[\mathbf{A/B}]_{\mathcal{P}_4}$, then it must be the case that the generalised paramodulation step was not necessary as the literal $[\mathbf{T}[\mathbf{A/B}]_{\mathcal{P}_2}]^F$ already is a flex-flex unification constraint and thus $\mathcal{C}_1[\mathbf{A/B}]_{\mathcal{P}} \equiv \square$, which trivially gives us the assertion.
- Without loss of generality let us assume that p' refers to a prefix of term $\mathbf{T}_1[\mathbf{A/B}]_{\mathcal{P}_3}$. We then know that $[\mathbf{T}_1[\mathbf{A/B}]_{\mathcal{P}_3} = \mathbf{T}_2[\mathbf{A/B}]_{\mathcal{P}_4}]^F$ must have form $[\mathbf{A}\ \overline{(\mathbf{U}[\mathbf{A/B}]^n_{\mathcal{P}_{4_n}})} = H\ \overline{\mathbf{V}[\mathbf{A/B}]^m_{\mathcal{P}_{5_m}}}]^F$ where $n \geq 0$ and $m \equiv 0$ (if $m > 0$ we cannot obtain a flex-flex constraint by the replacement). As the replacement of $\mathbf{A}$ by $\mathbf{B}$ introduces a flex-flex constraint we know that term $\mathbf{B}$ must have a flexible head, i.e. $\mathbf{B}$ has form $(F\ \overline{\mathbf{R}^l})$ such that $l+n > 0$. Now consider term $\mathbf{A}$: If the head of $\mathbf{A}$ is a variable then again the replacement of $\mathbf{A}$ is not necessary as we already have a flex-flex constraint without employing generalised paramodulation. Thus $\mathcal{C}_1[\mathbf{A/B}]_{\mathcal{P}} \equiv \square$ which trivially gives us the assertion. If on the other hand $\mathbf{A}$ has a rigid head, i.e. $\mathbf{A}$ has form $(a\ \overline{\mathbf{W}^k})$ for $k \geq 0$, then the proof is a bit more complicate as we have to construct a refutation using $\mathcal{C}_1[\mathbf{A/B}]_{\mathcal{P}}$ and $[\mathbf{A} = \mathbf{B}]^T$ without employing generalised paramodulation. Before discussing this derivation let us sum up the restrictions about the structure of clauses $\mathcal{C}_1[\mathbf{A/B}]_{\mathcal{P}}$ and $[\mathbf{A} = \mathbf{B}]^T$ in this case: The clause $\mathcal{C}_1[\mathbf{A/B}]_{\mathcal{P}}$ looks like $\mathcal{C}_1'[\mathbf{A/B}]_{\mathcal{P}_1} \vee [((a\ \overline{\mathbf{W}^k})\ \overline{(\mathbf{U}[\mathbf{A/B}]^n_{\mathcal{P}_{4_n}})}) = (H\ \overline{\mathbf{V}[\mathbf{A/B}]^m_{\mathcal{P}_{5_m}}})]^F$ and clause $[\mathbf{A} = \mathbf{B}]^T$ has form $[(a\ \overline{\mathbf{W}^k}) =^{\gamma_3\to o} (F\ \overline{\mathbf{R}^l})]^T$ where $a_{\gamma_1\to(\gamma_3\to o)}$ is a predicate constant and $F_{\gamma_2\to(\gamma_3\to o)}$ a predicate variable. We apply rule *FlexRigid* to the former clause and positive extensionality rule *Func'* for n-times to the latter clause.

$$\begin{aligned}
\mathcal{C}_2 &: \mathcal{C}_1'[\mathbf{A/B}]_{\mathcal{P}_1} \vee [((a\ \overline{\mathbf{W}^k})\ \overline{(\mathbf{U}[\mathbf{A/B}]^n_{\mathcal{P}_{4_n}})}) = (H\ \overline{\mathbf{V}[\mathbf{A/B}]^m_{\mathcal{P}_{5_m}}})]^\alpha \\
&\quad \vee [H = \lambda\overline{Z^m}.\ a\ \overline{(K\ \overline{Z^m})^{k+n}}]^F \\
\mathcal{C}_3 &: [(a\ \overline{\mathbf{W}^k})\ \overline{Y^n} = (F\ \overline{\mathbf{R}^l})\ \overline{Y^n}]^T
\end{aligned}$$

The variable K in $\mathcal{C}_2$ and the variables Y^n in $\mathcal{C}_3$ are new free variables. We substi-

tute the imitation binding in clause $\mathcal{C}_2$. (Thereby we assume that variable H does not occur in any of the literals in $\mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}_1}$. If on the other hand H occurs in $\mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}_1}$ then we only get a problem if H occurs at head position. We can then use an analogous argumentation to the following one for these literals as well.)
$\mathcal{C}_4 : \mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}_1} \vee [(a\ \overline{\mathbf{W}^k})\ \overline{(\mathbf{U}[\mathbf{A}/\mathbf{B}]^n_{\mathcal{P}_{4_n}})} = (a\overline{(K\ \overline{\mathbf{V}[\mathbf{A}/\mathbf{B}]^m_{\mathcal{P}_{5_m}}})^{k+n}})]^F$
and resolve[17] between $\mathcal{C}_4$ and $\mathcal{C}_3$ (note that that the unification terms of both unification constraints must have the same type)

$$\begin{aligned}\mathcal{C}_5 : \mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}_1} \vee [((a\ \overline{\mathbf{W}^k})\ \overline{(\mathbf{U}[\mathbf{A}/\mathbf{B}]^n_{\mathcal{P}_{4_n}})} &= (a\overline{(K\ \overline{\mathbf{V}[\mathbf{A}/\mathbf{B}]^m_{\mathcal{P}_{5_m}}})^{k+n}}))\\ = ((a\ \overline{\mathbf{W}^k})\ \overline{Y^n} =^o (F\ \overline{\mathbf{R}^l})\ \overline{Y^n})]^F\end{aligned}$$

We now apply the decomposition rule (2 times) and immediately replace the trivial pair $[===]^F$ with rule *Triv*.

$$\begin{aligned}\mathcal{C}_5 : \mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}_1} \vee [((a\ \overline{\mathbf{W}^k})\ \overline{(\mathbf{U}[\mathbf{A}/\mathbf{B}]^n_{\mathcal{P}_{4_n}})}) &= ((a\ \overline{\mathbf{W}^k})\ \overline{Y^n})]^F\\ \vee[(a\overline{(K\ \overline{\mathbf{V}[\mathbf{A}/\mathbf{B}]^m_{\mathcal{P}_{5_m}}})^{k+n}}) &= ((F\ \overline{\mathbf{R}^l})\ \overline{Y^n})]^F\end{aligned}$$

This latter two unification constraints are obviously solvable with substitution $[\overline{(\mathbf{U}[\mathbf{A}/\mathbf{B}]^n_{\mathcal{P}_{4_n}})}/\overline{Y^n}, \lambda\overline{Z^{l+n}}.\ (a\overline{(K\ \overline{\mathbf{V}[\mathbf{A}/\mathbf{B}]^m_{\mathcal{P}_{5_m}}})^{k+n}})/F]$ and as neither F nor the $\overline{Y^n}$ can occur in any other unification constraint in $\mathcal{C}_5$ we get that $\mathcal{C}_5 \vdash_{\mathcal{UNIF}} \square$ which proves the assertion.

$\underline{n > 1:}$ In the step case we consider the second derivation step (the first one following the generalised paramodulation step) in $\Delta_1 : \Phi * \mathcal{C}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}} * [\mathbf{A} = \mathbf{B}]^T \vdash^{GPara} \Phi * \mathcal{C}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}} * \mathcal{C}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}\setminus\{p\}} \vdash^r \Phi * \mathcal{C}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}} * \mathcal{C}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}\setminus\{p\}} * \mathcal{C}_2 \vdash_{\mathcal{ERUE}_k} \square$. If the premise clause(s) within the application of rule r are different from $\mathcal{C}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}\setminus\{p\}}$ we can obviously switch the first two derivation steps of Δ_1 such that we get the assertion trivially by employing the induction hypotheses. This even holds if the application of rule r indeed uses $\mathcal{C}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}\setminus\{p\}}$ but operates on a literal that was not affected by the initial generalised paramodulation step, i.e. on a literal different from the position where p refers to. Thus, for all of the following cases let us assume that clause $\mathcal{C}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}\setminus\{p\}}$ is of form $[\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'\setminus\{p'\}}]^\alpha \vee \mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}''}$ (for respective positions P', P'' and the position p' (the rest list of p) specifying the subterm that was modified within the initial generalised paramodulation step) and that r operates in the second derivation step on literal $[\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'\setminus\{p'\}}]^\alpha$. Note that that under this assumption the non-modified original clause $\mathcal{C}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}}$ is of form $[\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'}]^\alpha \vee \mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}''}$.

$r \in \{Res, Fac\}$: As both rules do not depend on the term structure of the resolution or factorisation literals we can in both cases switch the initial derivation steps such the assertion easily follows by induction hypotheses. We will briefly illustrate this here for rule *Res*. The argumentation for *Fac* is analogous. For the resolution step let us assume that there is a clause $\mathcal{C}_2 \in \Phi$ that is of form $[\mathbf{T}_2]^\beta \vee C_2'$ and that

$$\frac{[\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'\setminus\{p'\}}]^\alpha \vee \mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}''} \quad [\mathbf{T}_2]^\beta \vee C_2'}{\mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}''} \vee C_2' \vee [(\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'\setminus\{p'\}}) = \mathbf{T}_2]^F}\ Res$$

Then an analogous resolution proof step is possible between $\mathcal{C}_2$ and the non-modified clause $[\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'}]^\alpha \vee \mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}''}$

$$\frac{[\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'}]^\alpha \vee \mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}''} \quad [\mathbf{T}_2]^\beta \vee C_2'}{\mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}''} \vee C_2' \vee [(\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'}) = \mathbf{T}_2]^F}\ Res$$

[17] We do not need to switch the latter unification constraint here as $\mathcal{C}_4$ and $\mathcal{C}_3$ already have the right constellation. But generally it might be necessary to employ the symmetry rule to clause $\mathcal{C}_3$.

We can obviously apply generalised paramodulation rule *GPara* to the latter clause (at position p' in the new unification constraint) thereby generating exactly the same clause as in the above original derivation.

$$\frac{\mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}''} \vee C_2' \vee [\mathbf{T}_2 = (\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'})]^F \quad [\mathbf{A} = \mathbf{B}]^T}{\mathcal{C}_1'[\mathbf{A}/\mathbf{B}]_{\mathcal{P}''} \vee C_2' \vee [(\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'\setminus\{p'\}}) = \mathbf{T}_2]^F}\ GPara$$

Now the assertion follows by induction hypothesis.

$r \in \{Prim, Func', Equiv'\}$: The three cases are analogous and therefore we only discuss rule *Prim* here. Thereby we consider the position p' of the subterm that has been rewritten in the generalised paramodulation step. (i) If the position p' in literal $[\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'\setminus\{p'\}}]^\alpha$ refers to a proper subterm then we get the assertion analogously to the cases for $r \in \{Res, Fac\}$ above by switching the first two derivation steps and employing the induction hypotheses. (ii) In the other case position p' refers to a flexible prefix term of $\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'\setminus\{p'\}}$, i.e.

$$[\overbrace{(\ldots(\ldots(H\ \mathbf{U}_1)\ldots\mathbf{U}_k)}^{\mathbf{A}}\ldots\mathbf{U}_n)]^\alpha \qquad (*)$$

for $0 \leq k \leq n$. In case $k \equiv n$ we can replace the initial paramodulation step in Δ_1 by an alternative one not employing generalised paramodulation by (1a) and get the assertion by induction hypothesis. In case $k \leq n$ we first apply positive extensionality rule *Func'* for $(k-n)$-times to the clause $[\mathbf{A} = \mathbf{B}]^T$ leading to $[\mathbf{A}\ \overline{Y^{k-n}} = \mathbf{B}\ \overline{Y^{k-n}}]^T$. This time we can replace the initial paramodulation step in Δ_1 by an alternative one by (1b) and then employ induction hypothesis.

$r \in \{Subst, Triv, Func, Dec, FlexRigid, FlexFlex, Equiv, Leib\}$: If position p' in literal $[\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'\setminus\{p'\}}]^\alpha$ refers to a flexible prefix term as illustrated in (*) above we get the assertion analogously to above either immediately by (1a) and induction hypotheses, or by (1b) in combination with an appropriate modification of clause $[\mathbf{A} = \mathbf{B}]^T$ with rule *Func'*. If on the other hand position p refers to a proper subterm of $[\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'\setminus\{p'\}}]^\alpha$ then we get the result by employing derivation that is analogous to one already employed in remark 5.2.

$r \equiv Cnf$: We again differentiate between the following two cases: (i) position p' in $[\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'\setminus\{p'\}}]^\alpha$ refers to proper subterm of the literals atom and (ii) position p' in $[\mathbf{T}_1[\mathbf{A}/\mathbf{B}]_{\mathcal{P}'\setminus\{p'\}}]^\alpha$ refers to a prefix term of the atom (see (*) above). In case (i) the assertion follows by induction hypothesis after switching the first two derivation steps. This is possible as derivation step r is obviously also applicable to the initial clause first such that an subsequent application of rule *Para* leads to $\mathcal{C}_2$. In case (ii) the argumentation is analogous to the case (ii) for $r \in \{Prim, Func', Equiv'\}$ discussed above, i.e. we get the assertion by employing either (1a) and induction hypotheses or by (1b) in combination with an n-times application of rule *Func'* to literal $[\mathbf{A} = \mathbf{B}]^T$ and induction hypotheses.

□

Theorem 6.12 (Completeness of $\mathcal{ERUE}_{fc}$). *The calculus $\mathcal{ERUE}_{fc}$ is complete with respect to Henkin models.*

Proof: Let I_Σ be the set of Σ-sentences which cannot be refuted by the calculus $\mathcal{ERUE}_{fc}$ ($\mathrm{I}_\Sigma := \{\Phi \subseteq cwff_o(\Sigma) | \Phi_{cl} \not\vdash_{\mathcal{ERUE}_{fc}} \square\}$), then we show that I_Σ is a saturated abstract consistency class for Henkin models with primitive equality 2.3. This entails completeness with the model existence theorem for Henkin models with primitive equality 2.1(2).

First we have to verify that I_Σ validates the abstract consistency properties ∇_c, $\nabla_\neg$, ∇_β, $\nabla_\vee$, $\nabla_\wedge$, $\nabla_\forall$, $\nabla_\exists$, $\nabla_\mathfrak{b}$, $\nabla_\mathfrak{q}$ and that I_Σ is saturated. For all of these cases the proofs are identical to

the corresponding argumentations in theorem 3.16. The only difference is that we employ the lemmata 6.9(1)-(2) and 6.7 instead of 3.15(1)-(2) and 3.12. Thus, all we need to ensure is the validity of the additional abstract consistency property $\nabla_{\mathfrak{e}}$ for primitive equality.

$\nabla_{\mathfrak{e}}$ (r) $\neg(\mathbf{A} =^{\alpha} \mathbf{A}) \notin \Phi$
(s) if $\mathbf{F}[\mathbf{A}]_p \in \Phi$ and $\mathbf{A} = \mathbf{B} \in \Phi$, then $\Phi * \mathbf{F}[\mathbf{B}]_p \in \Gamma_\Sigma$

(r) We have that $[\mathbf{A} =^{\alpha} \mathbf{A}]^F \vdash^{Triv} \square$ and thus $\neg(\mathbf{A} =^{\alpha} \mathbf{A}) \notin \Phi$.
(s) Analogously to 5.20 we show the contrapositive of the assertion and thus we assume that there is derivation $\Delta_1 : \Phi_{cl} * [\mathbf{F}[\mathbf{B}]_p]^T \vdash_{\mathcal{ERUE}_{fc}} \square$. Now consider the following $\mathcal{ERUE}_{fc}$-derivation:

$$\frac{[\mathbf{F}[\mathbf{A}]_p]^T \quad [\mathbf{A} = \mathbf{B}]^T}{[\mathbf{F}[\mathbf{B}]_p]^T}\ GRue$$

By lemma 6.11(2) we know that the generalised paramodulation rule *GPara* is admissible for calculus $\mathcal{ERUE}_{fc}$ and thus there is a $\mathcal{ERUE}_{fc}$-derivation $\{[\mathbf{F}[\mathbf{A}]_p]^T, [\mathbf{A} = \mathbf{B}]^T\} \vdash_{\mathcal{ERUE}_{fc}} [\mathbf{F}[\mathbf{B}]_p]^T$. Consequently there is a derivation $\Delta_1 : \Phi * [\mathbf{F}[\mathbf{A}]_p]^T * [\mathbf{A} = \mathbf{B}]^T \vdash_{\mathcal{ERUE}_{fc}} \Phi_{cl}\Phi * [\mathbf{F}[\mathbf{A}]_p]^T * [\mathbf{A} = \mathbf{B}]^T * [\mathbf{F}[\mathbf{B}]_p]^T \vdash_{\mathcal{ERUE}_{fc}} \square$ which completes the proof. □

6.5 Theorem Equivalence

Analogously to subsections 3.5 and 5.6 we now prove that $\mathcal{ERUE}_{fc}$ and $\mathcal{ERUE}_f$ are theorem equivalent. Theorem equivalence of $\mathcal{ERUE}_{fc}$ and $\mathcal{ERUE}$ is then presented as a conjecture that has not been formally proven yet.

Lemma 6.13 (Proper Clauses in $\mathcal{ERUE}$ and $\mathcal{ERUE}_f$). *For each non-proper clause $\mathcal{C}$ and clause set Φ such that $\Phi* \vdash_{\mathcal{ERUE}_{fc}} \mathcal{C}$ we have that $\Phi \vdash_{\mathcal{ERUE}_f} \mathcal{C}$.*

Proof: Analogously to lemmata 3.20 and 5.21. Resolution on unification constraints does not cause any additional problems. □

As $\square$ is also a proper clause we immediately get the following corollary:

Corollary 6.14 (Theorem Equivalence of $\mathcal{ERUE}_{fc}$ and $\mathcal{ERUE}_f$). *The calculi $\mathcal{ERUE}_{fc}$ and $\mathcal{ERUE}_f$ are theorem equivalent.*

Conjecture 6.15 (Theorem Equivalence of $\mathcal{ERUE}$ and $\mathcal{ERUE}_{fc}$ (or $\mathcal{ERUE}_f$)). *The calculi $\mathcal{ERUE}$ and $\mathcal{ERUE}_{fc}$ (or $\mathcal{ERUE}_f$) are theorem equivalent.*

Again the author expects that the proof for this conjecture will be analogous to the proof of the theorem equivalence for calculi $\mathcal{ER}_{fc}$ and $\mathcal{ER}$ (or $\mathcal{EP}_{fc}$ and $\mathcal{EP}$) and thus this conjecture gains evidence by the examples already carried out with the LEO-prover [BK98b] for extensional higher-order resolution as well as the new challenging examples discussed for extensional higher-order paramodulation discussed those for extensional higher-order paramodulation discussed in section 7. There is example known to the author that demonstrates the necessity of rule *FlexFlex*.

7 Examples

This section presents a couple of interesting examples that illustrate the basic ideas of the calculi $\mathcal{ER}$, $\mathcal{EP}$ and $\mathcal{ERUE}$ discussed in this paper or even compare refutations in the different calculi with each other. Furthermore, nearly all examples demonstrate the importance of an appropriate extensionality treatment in a higher-order theorem prover.

7.1 Decomposition in $\mathcal{ER}$

This examples demonstrates the basic ideas of extensional higher-order resolution and illustrates that this approach can also be seen as test calculus for extensional higher-order E-unification.

Furthermore, it focuses on the role of the decomposition rule in connection with the extensionality rules and compares the slightly modified decomposition rule used in this paper with the rule *Dec'* used in [BK98a]:

$$\frac{\mathbf{C} \vee [h\ \overline{\mathbf{U}^n} = h\ \overline{\mathbf{V}^n}]^F}{\mathbf{C} \vee [\mathbf{U}^1 = \mathbf{V}^1]^F \vee \ldots \vee [\mathbf{U}^n = \mathbf{V}^n]^F}\ \mathit{Dec}'$$

Our example extends example **E1** from [BK98a]: Suppose we have four function constants ($f_{(\alpha\to\alpha)\to\alpha\to\alpha}$, $g_{(\alpha\to\alpha)\to\alpha\to\alpha}$, $h_{\alpha\to\alpha}$ and $j_{\alpha\to\alpha}$) and we know that f equals g and h equals j (in the E-unification perspective we can assume that this two equations define our theory E). We want to prove that under this assumptions (under this theory) $(f\ h)$ equals $(g\ j)$. Depending on the concrete encoding of the assumptions and the assertion this proof problem is either trivial or challenging for a automated theorem prover (as we may need to employ the extensionality principles). A challenging formulation that uses Leibniz equality is:

$$\mathbf{E}^{Dec}\ (\forall X_{\alpha\to\alpha}.\ \forall Y_{\alpha}.\ (f\ X\ Y) \doteq (g\ X\ Y)) \wedge (\forall Z_{\alpha}.\ (h\ Z) \doteq (j\ Z)) \Rightarrow (f\ h) \doteq (g\ j)$$

When expanding the definition of $\doteq$, negating the theorem and applying pre-clausification we obtain the following pre-clauses for this example:

$\mathcal{C}_1 : [\forall P_{\alpha\to o}.\ \forall X_{\alpha\to\alpha}.\ \forall Y_{\alpha}.\ \neg(P\ (f\ X\ Y)) \vee (P\ (g\ X\ Y))]^T$
$\mathcal{C}_2 : [\forall Q_{\alpha\to o} \forall Z_{\alpha}. .\ \neg(Q\ (h\ Z)) \vee (Q\ (j\ Z))]^T$
$\mathcal{C}_3 : [\neg(\forall R_{(\alpha\to\alpha)\to o}.\ (\neg(R\ (f\ h))) \vee (R\ (g\ j)))]^T$

By clause normalisation (rule *Cnf*) we get:

$\mathcal{C}_4 : [P\ (f\ X\ Y)]^F \vee [P\ (g\ X\ Y))]^T$ $\quad$ $\mathcal{C}_5 : [Q\ (h\ Z)]^F \vee [Q\ (j\ Z)]^T$
$\mathcal{C}_6 : [r\ (f\ h)]^T$ $\quad$ $\mathcal{C}_7 : [r\ (g\ j)]^F$

The reader may easily check that resolving $\mathcal{C}_6$ and $\mathcal{C}_7$ against $\mathcal{C}_4$ does not lead to successful proof attempt (c.f. discussion of example **E1** in [BK98a]).

A refutation is obviously possible with decomposition rule *Dec'*:

$\mathit{Res}(\mathcal{C}_6, \mathcal{C}_7)$:	$\mathcal{C}_8 : [r\ (f\ h) = r\ (g\ j)]^F$	
$\mathit{Dec}'(\mathcal{C}_8), \mathit{Triv}$:	$\mathcal{C}_9 : [f\ h = g\ j]^F$	
$\mathit{Dec}'(\mathcal{C}_9)$:	$\mathcal{C}_{10} : [f = g]^F \vee [h = j]^F$	
$2 \times \mathit{Func}(\mathcal{C}_{10})$:	$\mathcal{C}_{11} : [f\ t\ s = g\ t\ s]^F \vee [h\ u = j\ u]^F$	
$2 \times \mathit{Leib}(\mathcal{C}_{11})$:	$\mathcal{C}_{12} : [\forall P_{\alpha\to o}.\ \neg(P\ (f\ t\ s)) \vee (P\ (g\ t\ s))]^F \vee [\forall Q_{\alpha\to o}.\ \neg(Q\ (h\ u)) \vee (Q\ (j\ u))]^F$	
$\mathit{Cnf}(\mathcal{C}_{12})$:	$\mathcal{C}_{13} : [p\ (f\ t\ s)]^T \vee [q\ (h\ u)]^T$	$\mathcal{C}_{14} : [p\ (f\ t\ s)]^T \vee [q\ (j\ u)]^F$
	$\mathcal{C}_{15} : [p\ (g\ t\ s)]^F \vee [q\ (h\ u)]^T$	$\mathcal{C}_{16} : [p\ (g\ t\ s)]^F \vee [q\ (j\ u)]^F$
$\mathit{Prim}(\mathcal{C}_4)$:	$\mathcal{C}_{17} : [P'\ (f\ X\ Y)]^T \vee [P'\ (g\ X\ Y))]^F$	
$\mathit{Prim}(\mathcal{C}_5)$:	$\mathcal{C}_{18} : [Q'\ (h\ Z)]^T \vee [Q'\ (j\ Z)]^F$	

Within this derivation $t_{\alpha\to\alpha}, s_{\alpha}, u_{\alpha}, p_{\alpha\to o}$ and $q_{\alpha\to o}$ are new Skolem terms and $P'_{\alpha\to o}, Q'_{\alpha\to o}$ are new predicate variables. The rest of the refutation is now simple straightforward resolution between the clauses $\mathcal{C}_{13}$–$\mathcal{C}_{16}$ and our assumption clauses $\mathcal{C}_4, \mathcal{C}_5, \mathcal{C}_{17}$ and $\mathcal{C}_{18}$.

A alternative refutation that uses the rule *Dec* is a bit more tricky:

$\mathit{Res}(\mathcal{C}_6, \mathcal{C}_7)$:	$\mathcal{C}_8 : [r\ (f\ h) = r\ (g\ j)]^F$	
$\mathit{Dec}(\mathcal{C}_8)$:	$\mathcal{C}_9 : [f\ h = g\ j]^F$	
$\mathit{Func}(\mathcal{C}_9)$:	$\mathcal{C}_{10} : [f\ h\ s = g\ j\ s]^F$	
$\mathit{Leib}(\mathcal{C}_{10})$:	$\mathcal{C}_{11} : [\forall P_{\alpha\to o}.\ \neg(P\ (f\ h\ s)) \vee (P\ (g\ j\ s))]^F$	
$\mathit{Cnf}(\mathcal{C}_{11})$:	$\mathcal{C}_{12} : [p\ (f\ h\ s)]^T$	$\mathcal{C}_{13} : [p\ (g\ j\ s)]^F$
$\mathit{Res}(\mathcal{C}_{12}, \mathcal{C}_4)$:	$\mathcal{C}_{14} : [P\ (g\ X\ Y)]^T \vee [p\ (f\ h\ s) = P\ (f\ X\ Y)]^F$	
$\mathit{Res}(\mathcal{C}_{13}, \mathcal{C}_4)$:	$\mathcal{C}_{15} : [P'\ (f\ X\ Y))]^F \vee [p\ (g\ j\ s) = P'\ (g\ X\ Y)]^F$	

Note that the unification constraint in $\mathcal{C}_{14}$ as well as in $\mathcal{C}_{15}$ is solvable with an imitation and a projection solution and that this solutions can be computed by eager unifica-

tion with the pre-unification rules. In our proof we choose for $\mathcal{C}_{14}$ the "projection" solution $\sigma_1 := [\lambda X_\alpha.\, pX/P, h/X, s/Y]$ and for $\mathcal{C}_{15}$ the simple imitation solution $\sigma_2 := [\lambda X_\alpha.\, p\ (g\ j\ s)/P']$. We propagate this solutions back to the non-unification literals of the clauses and proceed as follows:

$$
\begin{array}{ll}
\mathcal{UNI}(\mathcal{C}_{14}): & \mathcal{C}_{16} : [p\ (g\ h\ s)]^T \\
\mathcal{UNI}(\mathcal{C}_{15}): & \mathcal{C}_{17} : [p\ (g\ j\ s)]^F \\
Res(\mathcal{C}_{16}, \mathcal{C}_{17}): & \mathcal{C}_{18} : [p\ (g\ h\ s) = p\ (g\ j\ s)]^F \\
2 \times Dec(\mathcal{C}_{18}): & \mathcal{C}_{19} : [h\ s = j\ s]^F \\
Leib(\mathcal{C}_{19}): & \mathcal{C}_{20} : [\forall Q_{\alpha\to o}.\, \neg(Q\ (h\ s)) \vee (Q\ (j\ s))]^F \\
Cnf(\mathcal{C}_{20}): & \mathcal{C}_{21} : [q\ (h\ s)]^T \qquad\qquad \mathcal{C}_{22} : [q\ (j\ s)]^T \\
Res(\mathcal{C}_{21}, \mathcal{C}_5): & \mathcal{C}_{23} : [Q\ (j\ Z)]^T \vee [q\ (h\ s) = Q\ (h\ Z)]^F \\
Res(\mathcal{C}_{22}, \mathcal{C}_5): & \mathcal{C}_{24} : [Q'\ (h\ Z')]^F \vee [q\ (j\ s) = Q'\ (j\ Z')]^F \\
Res(\mathcal{C}_{23}, \mathcal{C}_{24}): & [Q\ (j\ Z) = Q'\ (h\ Z')]^F \vee [q\ (h\ s) = Q\ (h\ Z)]^F \vee [q\ (j\ s) = Q'\ (j\ Z')]^F
\end{array}
$$

This unification constraint is obviously solvable with unifier $\sigma := [\lambda X_\alpha.\, q\ (h\ s)/Q, \lambda X_\alpha.\, q\ X/Q', s/Z']$ and this solution can be computed with the pre-unification rules.

When considering this examples the question arises whether rule *Dec* or rule rule *Dec'* is better suitable within calculus $\mathcal{ER}$. This problem should be examined and clarified by experiments in the LEO-prover. As an alternative to [BK98a] and in order to easify the proofs we decided in this paper to use *Dec*.

7.2 Leibniz Equality and Alternative Definitions in $\mathcal{ER}$

The examples discussed in this sub-section focus on the equivalence of Leibniz equality and alternative definitions for equality in higher-order logic.

The definitions for equality that are compared are:

Leibniz Equality $\doteq^\alpha := \lambda X_\alpha.\, \lambda Y_\alpha.\, \forall P_{\alpha\to o}.\, PX \Rightarrow PY$

Reflexivity Definition $\doteqdot^\alpha := \lambda X_\alpha.\, \lambda Y_\alpha.\, \forall Q_{\alpha\to\alpha\to o}.\, (\forall Z_\alpha.\, (Q\ Z\ Z)) \Rightarrow (Q\ X\ Y)$

Modified Leibniz Equality $\doteqdot^\alpha := \lambda X_\alpha.\, \lambda Y_\alpha.\, \forall P_{\alpha\to o}.\, ((a_o \vee \neg\, a_o) \wedge P\ X) \Rightarrow ((b_o \vee \neg\, b_o) \wedge P\ Y)$

Leibniz equality employs the substitutivity principle whereas the second alternative definition[18] employs the reflexivity principle. The third (artificial) definition illustrates that there are infinetely many analogous modifications of Leibniz equality (and also of the Reflexivity Definition) which in Henkin semantics all denote the same relation, namely equality. This examples furthermore illustrates that it is not possible to decide whether a given formula denotes the equality relation as this would require to decide the equivalence of two formulas.

$\mathbf{E}_1^{\doteq}$ **(i)** $(u \doteq^\alpha v) \Rightarrow (u \doteqdot^\alpha v)$ and **(ii)** $(u \doteqdot^\alpha v) \Rightarrow (u \doteq^\alpha v)$
In case (i) we use the following refutation ($q_{\alpha\to\alpha\to o}$ is a new Skolem constant):

$$
\begin{array}{ll}
\mathcal{CNF}(i): & \mathcal{C}_1 : [P\ u]^F \vee [P\ v]^T \\
 & \mathcal{C}_2 : [q\ Z\ Z]^T \\
 & \mathcal{C}_3 : [q\ u\ v]^F \\
Res(\mathcal{C}_1, \mathcal{C}_3): & \mathcal{C}_4 : [P\ u]^F \vee [P\ v = q\ u\ v]^F \\
\mathcal{UNI}(\mathcal{C}_4)\ mit\ \{\lambda X.\, q\ u\ X/P\}: & \mathcal{C}_5 : [q\ u\ v]^T \\
Res(\mathcal{C}_2, \mathcal{C}_5), Triv: & \square
\end{array}
$$

A refutation in case (ii) looks like ($p_{\alpha\to o}$ is a new Skolem constant):

[18] This definition is presented and discussed in Andrews textbook [And86] at page 155.

$\mathcal{CNF}(i)$:	$\mathcal{C}_1 : [Q\ z\ z]^F \vee [Q\ u\ v]^T$
	$\mathcal{C}_2 : [p\ u]^T$
	$\mathcal{C}_3 : [p\ v]^F$
$Prim(\mathcal{C}_1), Subst$:	$\mathcal{C}_4 : [Q'\ z\ z]^T \vee [Q'\ u\ v]^F$
$Res(\mathcal{C}_1, \mathcal{C}_3)$:	$\mathcal{C}_5 : [Q\ z\ z]^F \vee [Q\ u\ v = p\ v]^F$
$\mathcal{UNI}(\mathcal{C}_5)$ *mit* $\{\lambda X, Y.\ p\ Y/P\}$:	$\mathcal{C}_6 : [p\ z]^F$
$Res(\mathcal{C}_4, \mathcal{C}_2)$:	$\mathcal{C}_7 : [Q'\ z\ z]^T \vee [Q'\ u\ v = p\ u]^F$
$\mathcal{UNI}(\mathcal{C}_5)$ *mit* $\{\lambda X, Y.\ p\ X/P\}$:	$\mathcal{C}_8 : [p\ z]^T$
$Res(\mathcal{C}_6, \mathcal{C}_8), Triv$:	$\square$

$\mathbf{E}_2^{\doteq}$ **(i)** $(u \doteq^{\alpha} v) \Rightarrow (u \dddot{=}^{\alpha} v)$ and **(ii)** $(u \dddot{=}^{\alpha} v) \Rightarrow (u \doteq^{\alpha} v)$

In case (i) we proceed as follows:

$\mathcal{CNF}(i)$:	$\mathcal{C}_1 : [P\ u]^F \vee [P\ v]^T$
	$\mathcal{C}_2 : [p\ u]^T \vee [a]^T$
	$\mathcal{C}_3 : [p\ u]^T \vee [a]^F$
	$\mathcal{C}_4 : [p\ v]^F \vee [a]^T$
	$\mathcal{C}_5 : [p\ v]^F \vee [a]^F$
$Res(\mathcal{C}_2, \mathcal{C}_3), Fac, 2 \times Triv$:	$\mathcal{C}_6 : [p\ u]^T$
$Res(\mathcal{C}_4, \mathcal{C}_5), Fac, 2 \times Triv$:	$\mathcal{C}_7 : [p\ v]^T$
$Res(\mathcal{C}_1, \mathcal{C}_6)$:	$\mathcal{C}_8 : [P\ v]^T \vee [P\ u = pu]^F$
$Res(\mathcal{C}_8, \mathcal{C}_7)$:	$\mathcal{C}_9 : [P\ u = pu]^F \vee [P\ v = pv]^F$
$\mathcal{UNI}(\mathcal{C}_9)$ with $[p/P]$:	$\square$

For (ii) we have

$\mathcal{CNF}(i)$:	$\mathcal{C}_1 : [a]^F \vee [P\ u]^F \vee [b]^F \vee [P\ v]^T$
	$\mathcal{C}_2 : [a]^F \vee [P\ u]^F \vee [b]^T \vee [P\ v]^T$
	$\mathcal{C}_3 : [a]^T \vee [P\ u]^F \vee [b]^F \vee [P\ v]^T$
	$\mathcal{C}_4 : [a]^T \vee [P\ u]^F \vee [b]^T \vee [P\ v]^T$
	$\mathcal{C}_5 : [p\ u]^T$
	$\mathcal{C}_6 : [p\ v]^F$
$Res(\mathcal{C}_1, \mathcal{C}_2), 3 \times Fac, 4 \times Triv$:	$\mathcal{C}_7 : [a]^F \vee [P\ u]^F \vee [P\ v]^T$
$Res(\mathcal{C}_3, \mathcal{C}_4), 3 \times Fac, 4 \times Triv$:	$\mathcal{C}_8 : [a]^T \vee [P\ u]^F \vee [P\ v]^T$
$Res(\mathcal{C}_7, \mathcal{C}_8), 2 \times Fac, 3 \times Triv$:	$\mathcal{C}_9 : \vee [P\ u]^F \vee [P\ v]^T$
$Res(\mathcal{C}_9, \mathcal{C}_5)$:	$\mathcal{C}_{10} [P\ v]^T \vee [P\ u = pu]^F$
$Res(\mathcal{C}_{10}, \mathcal{C}_6)$:	$\mathcal{C}_{11} [P\ u = pu]^F \vee [P\ v = pv]^F$
$\mathcal{UNI}(\mathcal{C}_{11})$ with $[p/P]$:	$\square$

(Alternatively we can also prove that $\doteq^{\alpha} \doteq^{\alpha \to \alpha \to o} \dddot{=}^{\alpha}$) which is a bit more complicate.)

7.3 Positive Extensionality Rules in $\mathcal{EP}$ and $\mathcal{ERUE}$

The examples discussed here illustrate that the positive extensionality rules (or extensionality axioms which we want to avoid) are unavoidable in order to reach a Henkin completeness for our higher-order paramodulation approach $\mathcal{EP}$ and RUE-resolution approach $\mathcal{ERUE}$. As discussed in detail in section 4 none of these examples can be proven within naive higher-order paramodulation approach $\mathcal{EP}_{naive}$ (or in an analogous naive RUE-resoluttion approach).

$\mathbf{E}_1^{Para}$ $\mathcal{C}_1\ [a = \neg a]^T$ **Refutation in $\mathcal{EP}$ and $\mathcal{ERUE}$**

$Equiv'(\mathcal{C}_1), \mathcal{CNF}$:	$\mathcal{C}_2 : [A]^F \vee [A]^F$	$\mathcal{C}_3 : [A]^T \vee [A]^T$
$Fac(\mathcal{C}_2), \mathcal{UNI}$, $Fac(\mathcal{C}_3), \mathcal{UNI}$:	$\mathcal{C}_4 : [A]^F$	$\mathcal{C}_5 : [A]^T$
$Res(\mathcal{C}_4, \mathcal{C}_5), \mathcal{UNI}$:	$\mathcal{C}_6 : \square$	

Obviously paramodulation rule is not needed and thus this proof is also possible in calculus

$\mathcal{ERUE}$.

$\mathbf{E}_2^{Para}$ $\mathcal{C}_1 : [G\ X =^{\iota\to o} p]^T$ **Refutation in $\mathcal{EP}$ and $\mathcal{ERUE}$**

$Func'(\mathcal{C}_1)$:	$\mathcal{C}_2 : [G\ X\ Y =^o p\ Y]^T$	
$Equiv'(\mathcal{C}_1)$:	$\mathcal{C}_3 : [G\ X\ Y]^F \vee [p\ Y]^T$	$\mathcal{C}_4 : [G\ X\ Y]^T \vee [p\ Y]^F$
$Prim(\mathcal{C}_3), Subst$:	$\mathcal{C}_5 : [G'\ X\ Y]^T \vee [p\ Y]^T$	
$Prim(\mathcal{C}_4), Subst$:	$\mathcal{C}_6 : [G''\ X\ Y]^F \vee [p\ Y]^F$	
$Fac(\mathcal{C}_5), \mathcal{UNI}$:	$\mathcal{C}_7 : [p\ Y]^T$	
$Fac(\mathcal{C}_6), \mathcal{UNI}$:	$\mathcal{C}_8 : [p\ Y]^F$	
$Res(\mathcal{C}_7, \mathcal{C}_8), \mathcal{UNI}$:	$\mathcal{C}_9 : \square$	

Paramodulation rule is not needed and thus this proof is also possible in calculus $\mathcal{ERUE}$.

$\mathbf{E}_3^{Para}$ $\mathcal{C}_1 : [m = \lambda X_o.\ (\exists X_o.\ X \wedge \neg X)]^T$ **Refutation in $\mathcal{EP}$ and $\mathcal{ERUE}$**

$Func'(\mathcal{C}_1)$:	$\mathcal{C}_2 : [M\ Y_o = (\exists X_o.\ X \wedge \neg X)]^T$	
$Equiv'(\mathcal{C}_2)$:	$\mathcal{C}_3 : [M\ Y]^F \vee [s]^T$	$\mathcal{C}_4 : [M\ Y]^F \vee [s]^F$
$Prim(\mathcal{C}_3), Subst$:	$\mathcal{C}_5 : [H\ Y]^T \vee [s]^T$	
$Fac(\mathcal{C}_4), \mathcal{UNI}, Fac(\mathcal{C}_5), \mathcal{UNI}$:	$\mathcal{C}_6 : [s]^T$	$\mathcal{C}_7 : [s]^F$
$Res(\mathcal{C}_6, \mathcal{C}_7), \mathcal{UNI}$:	$\mathcal{C}_8 : \square$	

where s_o is a skolem constant for X. The paramodulation rule is not needed and thus this proof is also possible in calculus $\mathcal{ERUE}$.

$\mathbf{E}_4^{Para}$ $\mathcal{C}_1 : [m = \lambda X_o.\ \neg(m\ X)]^T$ **Refutation in $\mathcal{EP}$ and $\mathcal{ERUE}$**

$Func'(\mathcal{C}_1)$:	$\mathcal{C}_2 : [m\ Y = \neg(m\ Y)]^T$	
$Equiv'(\mathcal{C}_2)$:	$\mathcal{C}_3 : [m\ Y]^T \vee [m\ Y]^T$	$\mathcal{C}_4 : [m\ Y]^F \vee [m\ Y]^F$
$Fac(\mathcal{C}_3), \mathcal{UNI}, Fac(\mathcal{C}_4), \mathcal{UNI}$:	$\mathcal{C}_5 : [m\ Y]^T$	$\mathcal{C}_6 : [m\ Y]^F$
$Res(\mathcal{C}_5, \mathcal{C}_6), \mathcal{UNI}$:	$\mathcal{C}_7 : \square$	

The paramodulation rule is not needed and thus this proof is also possible in calculus $\mathcal{ERUE}$.

$\mathbf{E}_5^{Para}$ $\mathcal{C}_1 : [P\ q =^{\iota\to\iota\to\iota\to o} P\ r]^T$ $\mathcal{C}_2 : [q\ X =^{\iota\to o} \neg(r\ X)]^T$
where $P_{(\iota\to\iota\to o)\to(\iota\to\iota\to\iota\to o)}, X_\iota$ are free variables and $q_{\iota\to\iota\to o}, r_{\iota\to\iota\to o}$ are function constants.
Refutation in $\mathcal{EP}$

$3 \times Func'(\mathcal{C}_1)$:	$\mathcal{C}_3 : [P\ q\ Y_\iota^1\ Y_\iota^2\ Y_\iota^3 =^o P\ r\ Y_\iota^1\ Y_\iota^2\ Y_\iota^3]^T$
$Func'(\mathcal{C}_2)$:	$\mathcal{C}_4 : [q\ X\ Z_\iota =^o \neg(r\ X\ Z)]^T$
$Para(\mathcal{C}_4, \mathcal{C}_3)$:	$\mathcal{C}_5 : [P\ r\ Y_\iota^1\ Y_\iota^2\ Y_\iota^3]^T \vee [P\ q\ Y_\iota^1\ Y_\iota^2\ Y_\iota^3 =^o (q\ X\ Z =^o \neg(r\ X\ Z))]^F$

Now (pre)-unification applied to $\mathcal{C}_5$ is able to compute the following unifier for the unification constraint: $\{\lambda U_{\iota\to\iota\to o}, V_\iota, W_\iota, T_\iota.\ U\ V\ W =^o \neg(r\ V\ W)/P, V/Y^1, W/Y^2\}$. By eager substitution with rule *Subst* we therefore get:

$\mathcal{UNI}(\mathcal{C}_5), Subst$:	$\mathcal{C}_6 : [r\ X\ Z =^o \neg(r\ X\ Z)]^F$

The rest of the refutation is analogous to $\mathbf{E}_1^{Para}$.
Refutation in $\mathcal{ERUE}$

$3 \times Func'(\mathcal{C}_1)$:	$\mathcal{C}_3 : [P\ q\ Y_\iota^1\ Y_\iota^2\ Y_\iota^3 =^o P\ r\ Y_\iota^1\ Y_\iota^2\ Y_\iota^3]^T$	
$Func'(\mathcal{C}_2)$:	$\mathcal{C}_4 : [q\ X\ Z_\iota =^o \neg(r\ X\ Z)]^T$	
$Equiv'(\mathcal{C}_3)$:	$\mathcal{C}_5 : [P\ q\ Y_\iota^1\ Y_\iota^2\ Y_\iota^3]^F \vee [P\ r\ Y_\iota^1\ Y_\iota^2\ Y_\iota^3]^T$	
	$\mathcal{C}_6 : [P\ q\ Y_\iota^1\ Y_\iota^2\ Y_\iota^3]^T \vee [P\ r\ Y_\iota^1\ Y_\iota^2\ Y_\iota^3]^F$	
$Equiv'(\mathcal{C}_4), Cnf$:	$\mathcal{C}_7 : [q\ X\ Z_\iota]^F \vee [r\ X\ Z]^F$	$\mathcal{C}_8 : [q\ X\ Z_\iota]^T \vee [r\ X\ Z]^T$

The rest of the refutation is straightforward resolution on $\mathcal{C}_5, \ldots, \mathcal{C}_8$.

7.4 Comparing $\mathcal{EP}$ and $\mathcal{ERUE}$

7.4.1 Properties of Primitive Equality

In this examples we prove that primitive equality denotes an extensional congruence relation in all Henkin models (i.e. the intended equality relation). The single properties to be checked are reflexivity, symmetry, transitivity, congruence and extensionality.

$\mathbf{E}_r^=$ $\forall A_\alpha.\ A = A$. Clause normalisation leads to (a_α is a new Skolem constant):

$$\mathcal{C}_1 : [a = a]^F$$

Refutation in $\mathcal{EP}$: immediately by unification.
Refutation in $\mathcal{ERUE}$: immediately by unification.

$\mathbf{E}_s^=$ $\forall A_\alpha, B_\alpha.\ (A =^\alpha B) \Rightarrow (B =^\alpha A)$. Clause normalisation leads to ($a_\alpha, b_\alpha, p_{\alpha\to o}$ are new Skolem constants):

$$\mathcal{C}_1 : [a =^\alpha b]^T \qquad \mathcal{C}_2 : [b =^\alpha a]^F$$

Refutation in $\mathcal{EP}$:

$Leib(\mathcal{C}_2), \mathcal{CNF}$:	$\mathcal{C}_3 : [p\ a]^T$	$\mathcal{C}_4 : [p\ b]^F$
$Para(\mathcal{C}_1, \mathcal{C}_3), Triv$:	$\mathcal{C}_5 : [p\ b]^T$	
$Res(\mathcal{C}_4, \mathcal{C}_5), Triv$:	$\square$	

Refutation in $\mathcal{ERUE}$:

$Res(\mathcal{C}_1, \mathcal{C}_2), Triv$: $\square$ (possible beause of symmetry convention for unification constraints)

$\mathbf{E}_t^=$ $\forall A_\alpha, B_\alpha, C_\alpha.\ (A = B) \wedge (B = C) \Rightarrow (A = C)$. Clause normalisation leads to ($a_\alpha, b_\alpha, c_\alpha, p_{\alpha\to o}$ are new Skolem constants):

$$\mathcal{C}_1 : [a = b]^T \qquad \mathcal{C}_2 : [b = c]^T \qquad \mathcal{C}_3 : [a = c]^F$$

Refutation in $\mathcal{EP}$:

$Leib(\mathcal{C}_3), \mathcal{CNF}$:	$\mathcal{C}_4 : [p\ a]^T$	$\mathcal{C}_5 : [p\ c]^F$
$Para(\mathcal{C}_1, \mathcal{C}_4), Triv$:	$\mathcal{C}_6 : [p\ b]^T$	
$Para(\mathcal{C}_2, \mathcal{C}_6), Triv$:	$\mathcal{C}_7 : [p\ c]^T$	
$Res(\mathcal{C}_6, \mathcal{C}_7), Triv$:	$\square$	

Refutation in $\mathcal{ERUE}$:

$Res(c3, c2)$:	$\mathcal{C}_4 : [(a = c) = (b = c)]^F$
$2 \times Dec(c4), Triv$:	$\mathcal{C}_5 : [a = b]^F$
$Res(c1, c5), \mathcal{UNI}$:	$\mathcal{C}_6 : \square$

$\mathbf{E}_f^=$ $\forall F_{\alpha\to\beta}.\ \forall A_\alpha, B_\alpha.\ (A =^\alpha B) \Rightarrow (F\ A =^\beta F\ B)$. Clause normalisation leads to ($a_\alpha, b_\alpha, f_{\alpha\to\beta}, p_{\beta\to o}$ are new Skolem constants):

$$\mathcal{C}_1 : [a = b]^T \qquad \mathcal{C}_2 : [f\ a = f\ b]^F$$

Refutation in $\mathcal{EP}$:

$Leib(\mathcal{C}_2), \mathcal{CNF}$:	$\mathcal{C}_3 : [p\ (f\ a)]^T$	$\mathcal{C}_4 : [p\ (f\ b)]^F$
$Para(\mathcal{C}_1, \mathcal{C}_3), Triv$:	$\mathcal{C}_5 : [p\ (f\ b)]^T$	
$Res(\mathcal{C}_5, \mathcal{C}_4), Triv$:	$\square$	

Refutation in $\mathcal{ERUE}$:

$Dec(c2), Triv$:	$\mathcal{C}_3 : [a = b]^F$
$Res(c1, c3), \mathcal{UNI}$:	$\mathcal{C}_4 : \square$

$\mathbf{E}_e^=$ $\forall F_{\alpha\to\beta}, G_{\alpha\to\beta}.\ (\forall A_\alpha.\ (F\ A =^\beta G\ A) \Rightarrow (F = G)$. Clause normalisation leads to ($f_{\alpha\to\beta}, g_{\alpha\to\beta}, p_{\beta\to o}, s_\beta$ are new Skolem constants):

$$\mathcal{C}_1 : [f\ A = g\ A]^T \qquad \mathcal{C}_2 : [f = g]^F$$

Refutation in $\mathcal{EP}$:

$Func(\mathcal{C}_2)$:	$\mathcal{C}_3 : [f\ s = g\ s]^F$	
$Leib(\mathcal{C}_3), \mathcal{CNF}$:	$\mathcal{C}_4 : [p\ (f\ s)]^T$	$\mathcal{C}_5 : [p\ (g\ s)]^F$
$Para(\mathcal{C}_4, \mathcal{C}_1), \mathcal{UNI}$:	$\mathcal{C}_6 : [p\ (g\ s)]^T$	
$Res(\mathcal{C}_6, \mathcal{C}_5), \mathcal{UNI}$:	$\square$	

Refutation in $\mathcal{ERUE}$:

$Func(\mathcal{C}_2)$:	$\mathcal{C}_3 : [f\ s = g\ s]^F$
$Res(c1, c3), \mathcal{UNI}$:	$\mathcal{C}_4 : \square$

Especially $\mathbf{E}_e^=$ shows that in extensional higher-order paramodulation it might be useful to allow paramodulation also on unification constraints in order to get shorter and easier proofs whithout recursive calls from within the unification process. On the other hand allowing unification process to interact with unrestricted paramodulation and recursive calls to the overall proof search (which can still not be avoided generally) seems to be hardly tractable in practice.

7.4.2 Primitive Equality and Leibniz Equality

Here we analyze in our extensional higher-order paramodulation approach whether Leibniz equality and primitive equality denote the same relation. Analogous proofs are also possible in extensional higher-order RUE-resolution.

$\mathbf{E}_1^=$ We prove that $\doteq^\alpha = ^{\alpha\to\alpha\to o} = ^\alpha$, i.e. that $(\lambda X.\ \lambda Y.\ \forall P_{\alpha\to o}.\ P\ X \Rightarrow P\ Y) = ^{\alpha\to\alpha\to o} = ^\alpha$

Refutation in $\mathcal{EP}$ (u_α, v_α and $p_{\alpha\to o}, q_{\alpha\to o}, r_{\alpha\to o}$ are Skolem constants)

$\mathcal{CNF}(\mathbf{E}_1^=)$:	$\mathcal{C}_1 : [(\lambda X.\ \lambda Y.\ \forall P_{\alpha\to o}.\ P\ X \Rightarrow P\ Y) = ^{\alpha\to\alpha\to o} = ^\alpha]^F$
$2 \times Func(\mathcal{C}_1)$:	$\mathcal{C}_2 : [(\forall P_{\alpha\to o}.\ P\ u \Rightarrow P\ v) = ^o (u = ^\alpha v)]^F$
$Equiv(\mathcal{C}_2), \mathcal{CNF}$:	$\mathcal{C}_3 : [p\ u]^T \vee [u = ^\alpha v]^F$
	$\mathcal{C}_4 : [p\ u]^F \vee [u = ^\alpha v]^F$
	$\mathcal{C}_5 : [P\ v]^T \vee [u = ^\alpha v]^T \vee [P\ u]^F$
$Res(\mathcal{C}_3, \mathcal{C}_4), Triv$:	$\mathcal{C}_6 : [u = ^\alpha v]^F \vee [u = ^\alpha v]^F$
$2 \times Leib(\mathcal{C}_6)$:	$\mathcal{C}_7 : [\forall Q_{\alpha\to o}.\ Q\ u \Rightarrow Q\ v]^F \vee [\forall R_{\alpha\to o}.\ R\ u \Rightarrow R\ v]^F$
$\mathcal{CNF}(\mathcal{C}_7)$:	$\mathcal{C}_8 : [q\ u]^T \vee [r\ u]^T$
	$\mathcal{C}_9 : [q\ u]^T \vee [r\ v]^F$
	$\mathcal{C}_{10} : [q\ v]^F \vee [r\ u]^T$
	$\mathcal{C}_{11} : [q\ v]^F \vee [r\ v]^F$
$Prim(\mathcal{C}_5)$:	$\mathcal{C}_{12} : [P\ v]^T \vee [u = ^\alpha v]^T \vee [P\ u]^F$ $\vee [P = ^{\alpha\to o} \lambda X.\ (H_{\alpha\to\alpha}\ X) = ^\alpha (H'_{\alpha\to\alpha}\ X)]^F$
$Subst(\mathcal{C}_12)$:	$\mathcal{C}_{13} : [(H\ v) = ^\alpha (H'\ v)]^T \vee [u = ^\alpha v]^T \vee [(H\ u) = ^\alpha (H'\ u)]^F$
$Fac(\mathcal{C}_{13})$:	$\mathcal{C}_{14} : [u = ^\alpha v]^T \vee [(H\ u) = ^\alpha (H'\ u)]^F$ $\vee [((H\ v) = ^\alpha (H'\ v)) = ^o (u = ^\alpha v)]^F$
$2 \times Dec(\mathcal{C}_{14})$:	$\mathcal{C}_{15} : [u = ^\alpha v]^T \vee [(H\ u) = ^\alpha (H'\ u)]^F \vee [(H\ v) = u]^F$ $\vee [(H'\ v) = v]^F$
$2 \times FlexRigid(\mathcal{C}_{15})$:	$\mathcal{C}_{16} : [u = ^\alpha v]^T \vee [(H\ u) = ^\alpha (H'\ u)]^F \vee [(H\ v) = u]^F$ $\vee [(H'\ v) = v]^F \vee [H = \lambda X_\alpha.\ u]^F \vee [H = \lambda X_\alpha.\ X]^F$
$2 \times Subst(\mathcal{C}_{16})$:	$\mathcal{C}_{17} : [u = ^\alpha v]^T \vee [u = ^\alpha u]^F \vee [u = u]^F \vee [v = v]^F$ $\vee [H = \lambda X_\alpha.\ u]^F \vee [H' = \lambda X_\alpha.\ X]^F$
$2 \times Triv(\mathcal{C}_{17})$:	$\mathcal{C}_{18} : [u = ^\alpha v]^T$
$4 \times Para(\{\mathcal{C}_8, \ldots, \mathcal{C}_{11}\}, \mathcal{C}_{18}), Triv$:	

$\mathcal{C}_{19} : [q\ v]^T \vee [r\ u]^T$ $\mathcal{C}_{20} : [q\ v]^T \vee [r\ v]^F$ $\mathcal{C}_{21} : [q\ v]^F \vee [r\ v]^T$ $\mathcal{C}_{22} : [q\ v]^F \vee [r\ v]^F$

Straightforward Resolution on $\mathcal{C}_{19}, \ldots, \mathcal{C}_{22} : \square$

Refutation in $\mathcal{ERUE}$

We replay the derivation above and instead of the paramodulation steps between $\mathcal{C}_8, \ldots, \mathcal{C}_{11}$ and $\mathcal{C}_{18}$ we resolve between $\mathcal{C}_{18}$ and the unification constraint $\mathcal{C}_6$.

$\mathbf{E}_2^=$ We prove that $\doteq^\alpha \doteq^{\alpha\to\alpha\to o} = ^\alpha$, i.e. that $\forall Q_{(\alpha\to\alpha\to o)\to o}.\ (Q\ (\lambda X.\ \lambda Y.\ \forall P_{\alpha\to o}.\ P\ X \Rightarrow$

$P\ Y)) \Rightarrow (Q\ =^{\alpha})$

Refutation in $\mathcal{EP}$ ($q_{(\alpha\to\alpha\to o)\to o}$ is a Skolem constant):

$\mathcal{CNF}(\mathbf{E}_2^{=}):$ $\mathcal{C}_1 : [q\ (\lambda X.\ \lambda Y.\ \forall P_{\alpha\to o}.\ P\ X \Rightarrow P\ Y)]^T$
$\mathcal{C}_2 : [q\ =^{\alpha}]^F$
$Res(\mathcal{C}_1,\mathcal{C}_2):$ $\mathcal{C}_3 : [(q\ (\lambda X.\ \lambda Y.\ \forall P_{\alpha\to o}.\ P\ X \Rightarrow P\ Y)) =^{\alpha\to\alpha\to o} (q\ =^{\alpha})]^F$
$Dec(\mathcal{C}_3), Triv:$ $\mathcal{C}_4 : [(\lambda X.\ \lambda Y.\ \forall P_{\alpha\to o}.\ P\ X \Rightarrow P\ Y) =^{\alpha\to\alpha\to o} =^{\alpha}]^F$
$\mathcal{C}_4$ is identical to $\mathcal{C}_1$ in the corresponding refutation for $\mathbf{E}_1^{=}$ above : ...$\square$

Refutation in $\mathcal{ERUE}$
Analogous to the $\mathcal{ERUE}$-refutation for $\mathbf{E}_1^{=}$.

7.4.3 Reasoning about Sets

$\mathbf{E}_1^{set}$ If the set of odd numerals is defined as the set of non-even numerals then the powerset of the set of odd numerals greater than 100 is equal to the powerset of the set of even numerals greater that 100:

$$\{X|\ odd\ X \wedge num\ X\} = \{X|\ \neg\ ev\ X \wedge num\ X\} \Rightarrow 2^{\{X|\ odd\ X \wedge num\ X \wedge X>100\}} = 2^{\{X|\ \neg\ ev\ X \wedge num\ X \wedge X>100\}}$$

where the powerset is defined by $\lambda M_{\alpha\to o}.\ \lambda N_{\alpha\to o}.\ \forall X_{\alpha}.\ N\ X \Rightarrow M\ X$. Clause normalisation leads to:

$\mathcal{C}_1 : [\lambda X.\ (odd\ X \wedge num\ X) = \lambda X.\ (\neg\ ev\ X \wedge num\ X)]^T$
$\mathcal{C}_2 : [(\lambda N.\ \forall X.\ N\ X \Rightarrow ((odd\ X \wedge num\ X) \wedge X > 100)) =$
$(\lambda N.\ \forall X.\ N\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)))]^F$

A short refutation in paramodulation calculus $\mathcal{EP}$ looks like:

$Func'(\mathcal{C}_1):$ $\mathcal{C}_3 : [(odd\ Y \wedge num\ Y) = (\neg\ ev\ Y \wedge num\ Y)]^F$
$Func(\mathcal{C}_2):$ $\mathcal{C}_4 : [(\forall X.\ n\ X \Rightarrow ((odd\ X \wedge num\ X) \wedge X > 100))$
$= (\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)))]^F$
$Equiv(\mathcal{C}_4):$ $\mathcal{C}_5 : [\forall X.\ n\ X \Rightarrow ((odd\ X \wedge num\ X) \wedge X > 100)]^T \vee$
$[\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)]^T$
$\mathcal{C}_6 : [\forall X.\ n\ X \Rightarrow ((odd\ X \wedge num\ X) \wedge X > 100)]^F \vee$
$[\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)]^F$
$Para(\mathcal{C}_5,\mathcal{C}_3), Triv:$ $\mathcal{C}_7 : [\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)]^T \vee$
$[\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)]^T$
$Fac(\mathcal{C}_7), Triv:$ $\mathcal{C}_8 : [\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)]^T$
$Para(\mathcal{C}_6,\mathcal{C}_3), Triv:$ $\mathcal{C}_9 : [\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)]^F \vee$
$[\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)]^F$
$Fac(\mathcal{C}_9), Triv:$ $\mathcal{C}_{10} : [\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)]^F$
$Res(\mathcal{C}_8,\mathcal{C}_{10}), Triv:$ $\square$

The $\mathcal{ERUE}$-refutation seems to have a more goal directed character and is at least not more complicate than the above paramodulation proof:

$Func'(\mathcal{C}_1):$ $\mathcal{C}_3 : [(odd\ Y \wedge num\ Y) = (\neg\ ev\ Y \wedge num\ Y)]^1$
$Func(\mathcal{C}_2):$ $\mathcal{C}_4 : [(\forall X.\ n\ X \Rightarrow ((odd\ X \wedge num\ X) \wedge X > 100))$
$= (\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)))]^F$
$Dec(\mathcal{C}_4), Triv:$ $\mathcal{C}_5 : [(\lambda X.\ n\ X \Rightarrow ((odd\ X \wedge num\ X) \wedge X > 100))$
$= (\lambda X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)))]^F$
$Func(\mathcal{C}_5):$ $\mathcal{C}_6 : [(n\ s \Rightarrow ((odd\ s \wedge num\ s) \wedge s > 100))$
$= (n\ s \Rightarrow ((\neg\ ev\ s \wedge num\ s) \wedge s > 100)))]^F$
$3 \times Dec(\mathcal{C}_6), Triv:$ $\mathcal{C}_7 : [(odd\ s \wedge num\ s) = (\neg\ ev\ s \wedge num\ s)]^F$
$Res(\mathcal{C}_7,\mathcal{C}_3), \mathcal{UNI}:$ $\square$

If we slightly modify the example by switching the first two conjuncts in the definition of odd numerals (i.e. $\mathcal{C}_1 : [\lambda X.\ (num\ X \wedge odd\ X) = \lambda X.\ (\neg\ ev\ X \wedge num\ X)]^T$ and $\mathcal{C}_3 : [(odd\ X \wedge num\ X) = (\neg\ ev\ X \wedge num\ X)]^T$), then both refutations obviously needs to employ additional recursive calls from within unification in order to show that $[(num\ X \wedge odd\ X) = (odd\ X \wedge num\ X)]^F$ is a solvable extensional unification problem. More precisely in an analogous paramodulation refutation additional recursive calls are necessary to justify the paramodulation steps leading to $\mathcal{C}_8$ and $\mathcal{C}_{10}$. These clauses would look like $\mathcal{C}_8 : [\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)]^T \vee [(num\ X \wedge odd\ X) = (odd\ X \wedge num\ X)]^F$ and $\mathcal{C}_{10} : [\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge num\ X) \wedge X > 100)]^F \vee [(num\ X \wedge odd\ X) = (odd\ X \wedge num\ X)]^F$ and in both cases the unification constraints can be eliminated by recursive calls to overall proof procedure. In an RUE-resolution refutation that is analogous to the above one the last derivation step would result in $Res(\mathcal{C}_5, \mathcal{C}_3) : \mathcal{C}_7 : [(odd\ s \wedge num\ s) = (\neg\ ev\ s \wedge num\ s) = (num\ s \wedge odd\ s) = (\neg\ ev\ s \wedge num\ s)]^F$ such that decomposition and subsequent elimination of trivial pairs leads to $\mathcal{C}_8 : [(num\ X \wedge odd\ X) = (odd\ X \wedge num\ X)]^F$. Then an recursive call to the overall proof procedure leads to the refutation.

Our examples above demonstrates that a refutation based on a term-rewriting idea most likely needs to perform recursive calls that are itself based on the difference reduction idea in case syntactical unification is not strong enough to justify the particular rewrite steps. Thus in practice the paramodulation approach needs both: appropriate heuristics in order to guide a proof along the term-rewriting idea and appropriate heuristics that guide the side computations based on the difference reduction idea.

The following example demonstrates that in some situations the term-rewriting idea even seems to be completely inappropriate. The problem in this example which again only slightly modifies example $\mathbf{E}_1^{set}$ above is that the term structure of the involved terms does not allow the positive equation to rewrite at the *right* subterms. This example additionally illustrates that an efficient paramodulation approach is probably not easy to mechanize in practice (even if suitable reduction orderings are available).

$\mathbf{E}_2^{set}$ We now artificially complicate problem $\mathbf{E}_1^{set}$ by switching the conjuncts in the conclusion such that the term-rewriting idea seems to completely inappropriate. Instead we obviously have to employ difference reduction as main proof idea. Our sligthly modified problem is given by:

$$\{X|\ odd\ X \wedge num\ X\} = \{X|\ \neg\ ev\ X \wedge num\ X\} \Rightarrow 2^{\{X|\ (odd\ X \wedge X > 100) \wedge num\ X\}} = 2^{\{X|\ (\neg\ ev\ X \wedge X > 100) \wedge num\ X\}}$$

Clause normalisation leads to:

$$\begin{aligned}&\mathcal{C}_1 : [\lambda X.\ (odd\ X \wedge num\ X) = \lambda X.\ (\neg\ ev\ X \wedge num\ X)]^T\\ &\mathcal{C}_2 : [(\lambda N.\ \forall X.\ N\ X \Rightarrow ((odd\ X \wedge X > 100) \wedge num\ X))\\ &\quad = (\lambda N.\ \forall X.\ N\ X \Rightarrow ((\neg\ ev\ X \wedge X > 100) \wedge num\ X))]^F\end{aligned}$$

We proceed analogous to example $\mathbf{E}_1^{set}$:

$$\begin{aligned}Func'(\mathcal{C}_1) :&\quad \mathcal{C}_3 : [(odd\ X \wedge num\ X) = (\neg\ ev\ X \wedge num\ X)]^T\\ Func(\mathcal{C}_2) :&\quad \mathcal{C}_4 : [(\forall X.\ n\ X \Rightarrow ((odd\ s \wedge s > 100) \wedge num\ s)) =\\ &\qquad\qquad (\forall X.\ n\ X \Rightarrow ((\neg\ ev\ s \wedge s > 100) \wedge num\ s))]^F\end{aligned}$$

In we follow the paramodulation based refutation for $\mathbf{E}_1^{set}$ we would perform a recursive call to the refutation process with rule *Equiv* and then try to rewrite the resulting clauses:

$$\begin{aligned}Equiv(\mathcal{C}_4) :&\quad \mathcal{C}_5 : [\forall X.\ n\ X \Rightarrow ((odd\ X \wedge X > 100) \wedge num\ X)]^T \vee\\ &\qquad\qquad [\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge X > 100) \wedge num\ X)]^T\\ &\quad \mathcal{C}_6 : [\forall X.\ n\ X \Rightarrow ((odd\ X \wedge X > 100) \wedge num\ X)]^F \vee\\ &\qquad\qquad [\forall X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge X > 100) \wedge num\ X)]^F\end{aligned}$$

Sure there are many different rewwrite steps possible with $\mathcal{C}_3$, but as the reader may convince himself, none of these rewrite steps leads to unification constraints that are solvable (even recursive calls to the overall proof procedure cannot help and unfortunately only overwhelm the search space with useless clauses). The problem is that the *right* paramodulation step is not possible because of the unfortunately chosen term-structure (i.e. order of the conjuncts in the literals to be rewritten). The only way the author explored to find a refutation is to employ difference-reduction techniques as motivated by the corresponding RUE-resolution based derivation in example $\mathbf{E}_1^{set}$ above:

$$
\begin{array}{ll}
Dec(\mathcal{C}_4), Triv: & \mathcal{C}_5 : [(\lambda X.\ n\ X \Rightarrow ((odd\ X \wedge X > 100) \wedge num\ X)) \\
 & \qquad = (\lambda X.\ n\ X \Rightarrow ((\neg\ ev\ X \wedge X > 100) \wedge num\ X)))]^F \\
Func(\mathcal{C}_5): & \mathcal{C}_6 : [(n\ s \Rightarrow ((odd\ s \wedge s > 100) \wedge num\ s)) \\
 & \qquad = (n\ s \Rightarrow ((\neg\ ev\ s \wedge s > 100) \wedge num\ s)))]^F \\
Dec(\mathcal{C}_6), Triv: & \mathcal{C}_7 : [((odd\ s \wedge s > 100) \wedge num\ s) = ((\neg\ ev\ s \wedge s > 100) \wedge num\ s)]^F
\end{array}
$$

Instead of immediately resolving beetween $\mathcal{C}_7$ and $\mathcal{C}_3$ as in the corresponding derivation for $\mathbf{E}_1^{set}$ we employ the extensionality rules to both clauses and proceed with a straightforward resolution proof.

$$
\begin{array}{ll}
Equiv(\mathcal{C}_7)\mathcal{CNF}, & \\
Fac, \mathcal{UNI}: & \mathcal{C}_8 : [odd\ s]^T \vee [ev\ s]^F \\
 & \mathcal{C}_9 : [odd\ s]^T \vee [s > 100]^T \quad (subsumed\ by\ \mathcal{C}_{12}) \\
 & \mathcal{C}_{10} : [odd\ s]^T \vee [num\ s]^T \quad (subsumed\ by\ \mathcal{C}_{16}) \\
 & \mathcal{C}_{11} : [s > 100]^T \vee [ev\ s]^F \quad (subsumed\ by\ \mathcal{C}_{12}) \\
 & \mathcal{C}_{12} : [s > 100]^T \\
 & \mathcal{C}_{13} : [s > 100]^T \vee [num\ s]^T \quad (subsumed\ by\ \mathcal{C}_{12}) \\
 & \mathcal{C}_{14} : [num\ s]^T \vee [ev\ s]^F \quad (subsumed\ by\ \mathcal{C}_{16}) \\
 & \mathcal{C}_{15} : [num\ s]^T \vee [s > 100]^T \quad (subsumed\ by\ \mathcal{C}_{16}) \\
 & \mathcal{C}_{16} : [num\ s]^T \\
 & \mathcal{C}_{17} : [odd\ s]^F \vee [s > 100]^F \vee [num\ s]^F \vee [ev\ s]^T \\
Equiv'(\mathcal{C}_3), \mathcal{CNF}, & \\
Fac, \mathcal{UNI}: & \mathcal{C}_{18} : [odd\ X]^F \vee [num\ X]^F \vee [ev\ X]^F \\
 & \mathcal{C}_{19} : [odd\ X]^T \vee [num\ X]^F \vee [ev\ X]^T \\
Res(\mathcal{C}_{12}, \mathcal{C}_{17}), \mathcal{UNI}: & \mathcal{C}_{20} : [odd\ s]^F \vee [num\ s]^F \vee [ev\ s]^T \\
Res(\mathcal{C}_{16}, \mathcal{C}_{20}), \mathcal{UNI}: & \mathcal{C}_{21} : [odd\ s]^F \vee [ev\ s]^T \\
Res(\mathcal{C}_{16}, \mathcal{C}_{18}), \mathcal{UNI}: & \mathcal{C}_{22} : [odd\ X]^F \vee [ev\ X]^F \\
Res(\mathcal{C}_{16}, \mathcal{C}_{19}), \mathcal{UNI}: & \mathcal{C}_{23} : [odd\ X]^T \vee [ev\ X]^T
\end{array}
$$

Straightforward resolution refutation with $\mathcal{C}_8, \mathcal{C}_{21}, \mathcal{C}_{22}, \mathcal{C}_{23}$

An alternative Refutation in RUE-resolution approach that illustrates the difference-reduction idea even better is the following:

$$
\begin{array}{ll}
Res(\mathcal{C}_3, \mathcal{C}_7): & \mathcal{C}_5 : [((odd\ X \wedge num\ X) = (\neg\ ev\ X \wedge num\ X)) = \\
 & \qquad (((odd\ s \wedge s > 100) \wedge num\ s) = ((\neg\ ev\ s \wedge s > 100) \wedge num\ s)]^F \\
Equiv(\mathcal{C}_5), Cnf: & \\
\quad \mathcal{C}_7 : & [(odd\ X \wedge num\ X) = (\neg\ ev\ X \wedge num\ X)]^T \vee \\
 & \quad [((odd\ s \wedge s > 100) \wedge num\ s) = ((\neg\ ev\ s \wedge s > 100) \wedge num\ s)]^T \\
\quad \mathcal{C}_8 : & [(odd\ X \wedge num\ X) = (\neg\ ev\ X \wedge num\ X)]^F \vee \\
 & \quad [((odd\ s \wedge s > 100) \wedge num\ s) = ((\neg\ ev\ s \wedge s > 100) \wedge num\ s)]^F \\
2 \times Equiv'(\mathcal{C}_7), Cnf, Fac, Triv: & \\
\quad \mathcal{C}_9 : & [odd\ s]^T \vee [ev\ s]^T \vee [odd\ X]^T \vee [ev\ X]^T \vee [num\ s]^F \vee [s > 100]^F \vee [num\ X]^F \\
\quad \mathcal{C}_{10} : & [odd\ X]^T \vee [ev\ X]^T \vee [ev\ s]^F \vee [num\ s]^F \vee [s > 100]^F \vee [odd\ s]^F \vee [num\ X]^F \\
\quad \mathcal{C}_{11} : & [odd\ s]^T \vee [ev\ s]^T \vee [num\ s]^F \vee [s > 100]^F \vee [ev\ X]^F \vee [num\ X]^F \vee [odd\ X]^F \\
\quad \mathcal{C}_{12} : & [ev\ s]^F \vee [num\ s]^F \vee [s > 100]^F \vee [odd\ s]^F \vee [ev\ X]^F \vee [num\ X]^F \vee [odd\ X]^F
\end{array}
$$

$2 \times Equiv'(\mathcal{C}_7), Cnf, Fac, Triv:$
$\mathcal{C}_{13} : [num\ s]^T \vee [num\ X]^T$
$\mathcal{C}_{14} : [s > 100]^T \vee [num\ X]^T$
$\mathcal{C}_{15} : [odd\ s]^T \vee [num\ X]^T \vee [ev\ s]^F$
$\mathcal{C}_{16} : [s > 100]^T \vee [odd\ X]^T \vee [ev\ X]^F$
$\mathcal{C}_{17} : [odd\ s]^T \vee [odd\ X]^T \vee [ev\ s]^F \vee [ev\ X]^F$
$\mathcal{C}_{18} : [s > 100]^T \vee [ev\ X]^T \vee [odd\ X]^F \vee [num\ X]^F$
$\mathcal{C}_{19} : [ev\ s]^T \vee [num\ X]^T \vee [odd\ s]^F \vee [num\ s]^F \vee [s > 100]^F$
$\mathcal{C}_{20} : [ev\ s]^T \vee [odd\ X]^T \vee [odd\ s]^F \vee [num\ s]^F \vee [s > 100]^F \vee [ev\ X]^F$
$\mathcal{C}_{21} : [ev\ s]^T \vee [ev\ X]^T \vee [odd\ s]^F \vee [num\ s]^F \vee [s > 100]^F \vee [odd\ X]^F \vee [num\ X]^F$
$Straightforward\ resolution\ proof\ on\ \mathcal{C}_9, \ldots, \mathcal{C}_{21} : \square$

We could even resolve immediately between $\mathcal{C}_3$ and $\mathcal{C}_4$ and proceed with a recursive call to the overall proof procedure as illustrated above. The resulting set of clauses then again increases a bit but a straightforward resolution proof is still easy to find.

8 Conclusion

We presented the two approaches $\mathcal{EP}$ and $\mathcal{ERUE}$ for extensional higher-order paramodulation and RUE-resolution which extend the extensional higher-order resolution approach $\mathcal{ER}$ [BK98a] by a primitive equality treatment. All three approaches avoid the extensionality axioms and employ more goal directed extensionality rules instead. An interesting difference to Huet's original constraint resolution approach [Hue72] is that eager (pre-)unification becomes essential and is not generally delayable when an extensionality treatment is needed.

Henkin completeness has been proven for the slightly extended approaches $\mathcal{ER}_f$, $\mathcal{EP}_f$ and $\mathcal{ERUE}_f$, which additionally employ the guessing *FlexFlex* unification rule. Completeness of the pure approaches was presented only as a challenging claim and the crucial task will be to prove the admissibility of rule *FlexFlex*.

Furthermore, it has been motivated that some problems that require an extensionality treatment cannot be solved in the paramodulation approach $\mathcal{EP}$ by following the term-rewriting idea only. And it seems to be rather difficult to find suitable heuristics combining the term-rewriting and difference-reducing aspects of our approach in practice. As on the other hand the difference-reducing calculus $\mathcal{ERUE}$ seems to harmonise rather well with the (difference-reducing) extensionality rules (or axioms), this paper concludes with the question: Can HO adaptations of term-rewriting approaches keep their superiority (at least in many domains) known from FO if one is interested in Henkin completeness and extensionality, e.g. when reasoning about sets, where sets are encoded as characteristic functions? Further work will be to examine this aspect with the help of the *LEO*-system [BK98b] and to investigate the open questions of this paper.

Acknowledgments The work reported here was funded by the Deutsche Forschungsgemeinschaft under grant HOTEL. The author is grateful to M. Kohlhase, F. Pfenning, P. Andrews, V. Sorge and S. Autexier for stimulating discussions and support.

References

[AMS98] Serge Autexier, Heiko Mantel, and Werner Stephan. Simultaneous quantifier elimination. In O. Herzog and A. Günter, editors, *KI-98: Advances in Artificial Intelligence, 22nd Annual German Conference on Artificial Intelligence*, Bremen, Germany, 1998. Springer Verlag, LNCS 1504.

[And71] Peter B. Andrews. Resolution in type theory. *Journal of Symbolic Logic*, 36(3):414–432, 1971.

[And72] Peter B. Andrews. General models descriptions and choice in type theory. *Journal of Symbolic Logic*, 37(2):385–394, 1972.

[And86] Peter B. Andrews. *An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof*. Academic Press, 1986.

[Bar80] Hendrik P. Barendregt. *The Lambda-Calculus: Its Syntax and Semantics*. North-Holland, 1980.

[Bar84] H. P. Barendregt. *The Lambda Calculus*. North Holland, 1984.

[BCF+97] C. Benzmüller, L. Cheikhrouhou, D. Fehrer, A. Fiedler, X. Huang, M. Kerber, M. Kohlhase, K. Konrad, E. Melis, A. Meier, W. Schaarschmidt, J. Siekmann, and V. Sorge. ΩMEGA: Towards a mathematical assistant. In William McCune, editor, *Proceedings of the 14th Conference on Automated Deduction*, number 1249 in LNAI, pages 252–255, Townsville, Australia, 1997. Springer Verlag.

[Ben97] Christoph Benzmüller. A calculus and a system architecture for extensional higher-order resolution. Research Report 97-198, Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh,USA, June 1997.

[BGLS92] Leo Bachmair, Harald Ganzinger, Christopher Lynch, and Wayne Snyder. Basic paramodulation and superposition. In Kapur [Kap92], pages 66–78.

[BK97a] Christoph Benzmüller and Michael Kohlhase. Resolution for Henkin models. SEKI-Report SR-97-10, Universität des Saarlandes, 1997.

[BK97b] Christoph Benzmüuller and Michael Kohlhase. Model existence for higher-order logic. SEKI-Report SR-97-09, Universität des Saarlandes, 1997.

[BK98a] Christoph Benzmüller and Michael Kohlhase. Extensional higher order resolution. In Kirchner and Kirchner [KK98], pages 56–72.

[BK98b] Christoph Benzmüller and Michael Kohlhase. LEO, a higher order theorem prover. In Kirchner and Kirchner [KK98], pages 139–144.

[Byl89] Czeslaw Bylinski. Basic properties of sets. *Journal of Formalized Mathematics*, 1, 1989.

[Chu40] Alonzo Church. A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5:56–68, 1940.

[Dar68] J. L. Darlington. Automatic theorem porving with equality substitutions and mathematical induction. *Machine Intelligence*, 3:113–130, 1968.

[Dig79] Vincent J. Digricoli. Resolution by unificatoin and equality. In William H. Joyner, editor, *Proceedings of the 4th Workshop on Automated Deduction*, Austin, Texas, USA, 1979.

[GS89] Jean H. Gallier and Wayne Snyder. Complete sets of transformations for general E-unification. *Theoretical Computer Science*, 1(67):203–260, 1989.

[HBVL97] Th. Hillenbrand, A. Buch, R. Vogt, and B. Löchner. Waldmeister: High-performance equational deduction. *Journal of Automated Reasoning*, 18(2), 1997.

[Hen50] Leon Henkin. Completeness in the theory of types. *Journal of Symbolic Logic*, 15(2):81–91, 1950.

[Hue72] Gérard P. Huet. *Constrained Resolution: A Complete Method for Higher Order Logic*. PhD thesis, Case Western Reserve University, 1972.

[Hue73a] Gérard P. Huet. A mechanization of type theory. In Donald E. Walker and Lewis Norton, editors, *Proceedings of the 3rd International Joint Conference on Artificial Intelligence*, pages 139–146, 1973.

[Hue73b] Gérard P. Huet. The undecidability of unification in third order logic. *Information and Control*, 22(3):257–267, 1973.

[Hue75] Gérard P. Huet. An unification algorithm for typed λ-calculus. *Theoretical Computer Science*, 1:27–57, 1975.

[JP72] D. C. Jensen and Thomasz Pietrzykowski. A complete mechanization of ($Omega$)-order type theory. In *Proceedings of the ACM annual Conference*, volume 1, pages 82–92, 1972.

[JR96] Jean-Pierre Jouannaud and Albert Rubio. A recursive path ordering for higher-order terms in η-long β-normal form. In Harald Ganzinger, editor, *Proceedings of the 7th International Conference on Rewriting Techniques and Applications (RTA-96)*, volume 1103 of *LNCS*, pages 108–122, New Brunswick, NJ, USA, 1996. Springer-Verlag.

[JR98] Jean-Pierre Jouannaud and Albert Rubio. Rewrite orderings for higher-order terms in η-lon β-normal form and the recursive path ordering. *Theoretical Computer Science*, 1998. to appear.

[Kap92] D. Kapur, editor. *Proceedings of the 11th Conference on Automated Deduction*, volume 607 of *LNCS*, Saratoga Spings, NY, USA, 1992. Springer Verlag.

[KK98] Claude Kirchner and Helene Kirchner, editors. *Proceedings of the 15th Conference on Automated Deduction*, number 1421 in LNAI, Lindau, , Germany, 1998. Springer Verlag.

[Koh94] Michael Kohlhase. *A Mechanization of Sorted Higher-Order Logic Based on the Resolution Principle*. PhD thesis, Universität des Saarlandes, 1994.

[Koh95] Michael Kohlhase. Higher-Order Tableaux. In P. Baumgartner, R. Hähnle, and J. Posegga, editors, *Theorem Proving with Analytic Tableaux and Related Methods*, volume 918 of *Lecture Notes in Artificial Intelligence*, pages 294–309, 1995.

[LB98] Maxim Lifantsev and Leo Bachmeier. An lpo-based termination ordering for higher-order terms without λ-abstraction. In Jim Grundy and Malcolm Newey, editors, *Proceedings of the 11th International Conference Theorem Proving in Higher Order Logics (TPHOLs'98)*, volume 1479 of *LNCS*, pages 277–293, Canberra, Australia, 1998. Springer.

[Mil83] Dale Miller. *Proofs in Higher-Order Logic*. PhD thesis, Carnegie-Mellon University, 1983.

[Mil91] Dale Miller. A logic programming language with lambda-abstraction, function variables, and simple unification. *Journal of Logic and Computation*, 4(1):497–536, 1991.

[Mil92] Dale Miller. Unification under a mixed prefix. *Journal of Symbolic Computation*, 14:321–358, 1992.

[MW94] Olaf Müller and Franz Weber. Theory and practice of minimal modular higher-order E-unification. In Alan Bundy, editor, *Proceedings of the 12th Conference on Automated Deduction*, number 814 in LNAI, pages 650–677, Nancy, France, 1994. Springer Verlag.

[Nip93] Tobias Nipkow. Functional unification of higher-order patterns. In *Proceedings of the 8th Annual IEEE Symposium on Logic in Computer Science (LICS-8)*, Montreal, Canada, 1993. IEEE Computer Society Press.

[NQ91] Tobias Nipkow and Zhenyu Qian. Modular higher-order E-unification. In Ronald V. Book, editor, *Proceedings of the* 4^{th} *International Conference on Rewriting Techniques and Applications*, number 488 in LNCS, pages 200–214. Springer Verlag, 1991.

[NQ92] Tobias Nipkow and Zhenyu Qian. Reduction and unification in lambda calculi with subtypes. In Kapur [Kap92], pages 66–78.

[NR95] R. Nieuwenhuis and A. Rubio. Theorem proving with ordering and equality constrained clauses. *Journal of Symbolic Computation*, 19(4):321–352, 1995.

[Pie73] Thomasz Pietrzykowski. A complete mechanization of second-order type theory. *Journal of the Association for Computing Machinery*, 20:333–364, 1973.

[Pre98] Christian Prehofer. *Solving Higher-Order Equations: From Logic to Programming.* Progress in theoretical computer science. Birkhäuser, 1998.

[Rob65] J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the Association for Computing Machinery*, 12(1):23–41, 1965.

[RW69] Arthur Robinson and Larry Wos. Paramodulation and TP in first order theories with equality. *Machine Intelligence*, 4:135–150, 1969.

[Smu63] Raymond M. Smullyan. A unifying principle for quantification theory. *Proc. Nat. Acad Sciences*, 49:828–832, 1963.

[Sny90] Wayne Snyder. Higher order E-unification. In Mark Stickel, editor, *Proceedings of the 10th Conference on Automated Deduction*, number 449 in LNCS, pages 573–578, Kaiserslautern, Germany, 1990.

[Sny91] Wayne Snyder. *A Proof Theory for General Unification.* Progress in Computer Science and Applied Logic. Birkhäuser, 1991.

[TS89] Z. Trybulec and H. Swieczkowska. Boolean properties of sets. *Journal of Formalized Mathematics*, 1, 1989.

[Wei97] Christoph Weidenbach. SPASS: Version 0.49. *Journal of Automated Reasoning*, 18(2):247–252, 1997. Special Issue on the CADE-13 Automated Theorem Proving System Competition.

[Wol93] David A. Wolfram. *The Clausal Theory of Types.* Cambridge University Press, 1993.

[Wol94] David A. Wolfram. A semantics for λ-PROLOG. *Theoretical Computer Science*, 136(1), 1994.