Approaches to Higher-Order Resolution

Preliminaries and Notation

- only logical constants: $\neg_{o\to o}$, $\vee_{o\to o\to o}$, and $\Pi_{(\alpha\to o)\to o}$

Preliminaries and Notation

- only logical constants: $\neg_{o\rightarrow o}$, $\vee_{o\rightarrow o\rightarrow o}$, and $\Pi_{(\alpha\rightarrow o)\rightarrow o}$
- other logical operators can be defined (e.g., $\mathbf{A} \wedge \mathbf{B} := \neg(\neg\mathbf{A} \vee \neg\mathbf{B})$, $\forall X_\alpha.\mathbf{P}\ X := \Pi_{((\alpha\rightarrow o)\rightarrow o)}(\lambda X_\alpha.\mathbf{P}\ X)$, and $\exists X_\alpha.\mathbf{P}\ X := \neg\forall X_\alpha.\neg(\mathbf{P}\ X))$)

Preliminaries and Notation

- only logical constants: $\neg_{o\to o}$, $\vee_{o\to o\to o}$, and $\Pi_{(\alpha\to o)\to o}$
- other logical operators can be defined (e.g., $\mathbf{A} \wedge \mathbf{B} := \neg(\neg\mathbf{A} \vee \neg\mathbf{B})$, $\forall X_\alpha . \mathbf{P}\ X := \Pi_{((\alpha\to o)\to o)}(\lambda X_\alpha . \mathbf{P}\ X)$, and $\exists X_\alpha . \mathbf{P}\ X := \neg\forall X_\alpha . \neg(\mathbf{P}\ X))$)
- 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., $\mathbf{T}_\alpha$)

Preliminaries and Notation

- only logical constants: $\neg_{o \to o}$, $\vee_{o \to o \to o}$, and $\Pi_{(\alpha \to o) \to o}$
- other logical operators can be defined (e.g., $\mathbf{A} \wedge \mathbf{B} := \neg(\neg\mathbf{A} \vee \neg\mathbf{B})$, $\forall X_\alpha . \mathbf{P}\, X := \Pi_{((\alpha \to o) \to o)}(\lambda X_\alpha . \mathbf{P}\, X)$, and $\exists X_\alpha . \mathbf{P}\, X := \neg\forall X_\alpha . \neg(\mathbf{P}\, X)$))
- 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., $\mathbf{T}_\alpha$)
- we abbreviate function applications by $h_{\alpha_1 \to \dots \to \alpha_n \to \beta}\, \overline{\mathbf{U}^n_{\alpha_n}}$, which stands for $(\cdots(h_{\alpha_1 \to \dots \to \alpha_n \to \beta}\, \mathbf{U}^1_{\alpha_1}) \cdots \mathbf{U}^n_{\alpha_n})$.

Preliminaries and Notation

- only logical constants: $\neg_{o\to o}$, $\vee_{o\to o\to o}$, and $\Pi_{(\alpha\to o)\to o}$
- other logical operators can be defined (e.g., $\mathbf{A} \wedge \mathbf{B} := \neg(\neg\mathbf{A} \vee \neg\mathbf{B})$, $\forall X_\alpha . \mathbf{P}\ X := \Pi_{((\alpha\to o)\to o)}(\lambda X_\alpha . \mathbf{P}\ X)$, and $\exists X_\alpha . \mathbf{P}\ X := \neg\forall X_\alpha . \neg(\mathbf{P}\ X)$))
- 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., $\mathbf{T}_\alpha$)
- we abbreviate function applications by $h_{\alpha_1\to\cdots\to\alpha_n\to\beta}\ \overline{\mathbf{U}^n_{\alpha_n}}$, which stands for $(\cdots(h_{\alpha_1\to\cdots\to\alpha_n\to\beta}\ \mathbf{U}^1_{\alpha_1})\cdots\mathbf{U}^n_{\alpha_n})$.
- α-, β-, η-, $\beta\eta$-conversion and the definition of β-normal, $\beta\eta$-normal, long $\beta\eta$-normal, and head-normal form defined as usual (see [Barendregt84])

Preliminaries and Notation

- only logical constants: $\neg_{o\to o}$, $\vee_{o\to o\to o}$, and $\Pi_{(\alpha\to o)\to o}$
- other logical operators can be defined (e.g., $\mathbf{A} \wedge \mathbf{B} := \neg(\neg\mathbf{A} \vee \neg\mathbf{B})$, $\forall X_\alpha . \mathbf{P}\ X := \Pi_{((\alpha\to o)\to o)}(\lambda X_\alpha . \mathbf{P}\ X)$, and $\exists X_\alpha . \mathbf{P}\ X := \neg\forall X_\alpha . \neg(\mathbf{P}\ X))$)
- 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., $\mathbf{T}_\alpha$)
- we abbreviate function applications by $h_{\alpha_1\to\cdots\to\alpha_n\to\beta}\ \overline{\mathbf{U}^n_{\alpha_n}}$, which stands for $(\cdots(h_{\alpha_1\to\cdots\to\alpha_n\to\beta}\ \mathbf{U}^1_{\alpha_1})\cdots\mathbf{U}^n_{\alpha_n})$.
- α-, β-, η-, $\beta\eta$-conversion and the definition of β-normal, $\beta\eta$-normal, long $\beta\eta$-normal, and head-normal form defined as usual (see [Barendregt84])
- substitutions defined as usual

Preliminaries and Notation

- substitutions are represented as $[\mathbf{T}_1/\mathsf{X}_1, \ldots, \mathbf{T}_\mathsf{n}/\mathsf{X}_\mathsf{n}]$ where the X_i specify the variables to be replaced by the terms $\mathbf{T}_\mathsf{i}$. The application of a substitution σ to a term (resp. literal or clause) $\mathbf{C}$ is printed $\mathbf{C}_\sigma$

Preliminaries and Notation

- substitutions are represented as $[\mathbf{T}_1/\mathsf{X}_1, \ldots, \mathbf{T}_\mathsf{n}/\mathsf{X}_\mathsf{n}]$ where the X_i specify the variables to be replaced by the terms $\mathbf{T}_\mathsf{i}$. The application of a substitution σ to a term (resp. literal or clause) $\mathbf{C}$ is printed $\mathbf{C}_\sigma$
- a resolution calculus R provides a set of rules $\{\mathsf{r}_\mathsf{n} | \ 0 < \mathsf{n} \leq \mathsf{i}\}$ defined on clauses

Preliminaries and Notation

- substitutions are represented as $[\mathbf{T}_1/\mathsf{X}_1, \ldots, \mathbf{T}_\mathsf{n}/\mathsf{X}_\mathsf{n}]$ where the X_i specify the variables to be replaced by the terms $\mathbf{T}_\mathsf{i}$. The application of a substitution σ to a term (resp. literal or clause) $\mathbf{C}$ is printed $\mathbf{C}_\sigma$
- a resolution calculus R provides a set of rules $\{\mathsf{r}_\mathsf{n} \mid 0 < \mathsf{n} \leq \mathsf{i}\}$ defined on clauses
- we write $\Phi \vdash^{\mathsf{r}_\mathsf{n}} \mathcal{C}$ $(\mathcal{C}' \vdash^{\mathsf{r}_\mathsf{n}} \mathcal{C})$ iff clause $\mathcal{C}$ is the result of a one step application of rule $\mathsf{r}_\mathsf{n} \in \mathsf{R}$ to premise clauses $\mathcal{C}'_\mathsf{i} \in \Phi$ (to $\mathcal{C}'$ respectively)

Preliminaries and Notation

- substitutions are represented as $[\mathbf{T}_1/\mathsf{X}_1, \ldots, \mathbf{T}_n/\mathsf{X}_n]$ where the X_i specify the variables to be replaced by the terms $\mathbf{T}_i$. The application of a substitution σ to a term (resp. literal or clause) $\mathbf{C}$ is printed $\mathbf{C}_\sigma$
- a resolution calculus R provides a set of rules $\{r_n \mid 0 < n \leq i\}$ defined on clauses
- we write $\Phi \vdash^{r_n} \mathcal{C}$ ($\mathcal{C}' \vdash^{r_n} \mathcal{C}$) iff clause $\mathcal{C}$ is the result of a one step application of rule $r_n \in \mathsf{R}$ to premise clauses $\mathcal{C}'_i \in \Phi$ (to $\mathcal{C}'$ respectively)
- multiple step derivations in calculus R are abbreviated by $\Phi_1 \vdash_R \Phi_k$ (or $\mathcal{C}_1 \vdash_R \mathcal{C}_k$)

Def.: General Bindings

Let $\alpha := (\overline{\beta^{\mathsf{l}}} \longrightarrow \gamma)$ and let h be a constant or variable of type $(\overline{\delta_{\mathsf{m}}} \longrightarrow \gamma)$ in Σ,

Def.: General Bindings

Let $\alpha := (\overline{\beta^{\mathsf{l}}} \rightarrow \gamma)$ and let h be a constant or variable of type $(\overline{\delta_{\mathsf{m}}} \rightarrow \gamma)$ in Σ, then

$$\mathbf{G} := \lambda \overline{\mathsf{X}^{\mathsf{l}}_{\beta^{\mathsf{l}}}}.\mathsf{h}\ \overline{\mathbf{V}^{\mathsf{m}}}$$

($\mathsf{m} \geq 0$) is called a partial binding of type α and head h (see also [SnGa89,Snyder91]),

Def.: General Bindings

Let $\alpha := (\overline{\beta^l} \rightarrow \gamma)$ and let h be a constant or variable of type $(\overline{\delta_m} \rightarrow \gamma)$ in Σ, then

$$\mathbf{G} := \lambda \overline{X^l_{\beta^l}}.h\ \overline{\mathbf{V}^m}$$

($m \geq 0$) is called a partial binding of type α and head h (see also [SnGa89,Snyder91]), if $\mathbf{V}^i = H^i\ \overline{X^l_{\beta_l}}$ and the H^i are new variables of types $\overline{\beta^l} \rightarrow \delta^i$.

Def.: General Bindings

Let $\alpha := (\overline{\beta^l} \rightarrow \gamma)$ and let h be a constant or variable of type $(\overline{\delta_m} \rightarrow \gamma)$ in Σ, then

$$\mathbf{G} := \lambda \overline{X^l_{\beta^l}}.h\ \overline{\mathbf{V}^m}$$

($m \geq 0$) is called a partial binding of type α and head h (see also [SnGa89,Snyder91]), if $\mathbf{V}^i = H^i\ \overline{X^l_{\beta_l}}$ and the H^i are new variables of types $\overline{\beta^l} \rightarrow \delta^i$.

Partial bindings, where the head is a bound variable $X^j_{\beta_j}$ are called projection bindings (we write them as $\mathcal{G}^j_\alpha$) and imitation bindings (written $\mathcal{G}^h_\alpha$) otherwise.

Def.: General Bindings

Let $\alpha := (\overline{\beta^l} \rightarrow \gamma)$ and let h be a constant or variable of type $(\overline{\delta_m} \rightarrow \gamma)$ in Σ, then

$$\mathbf{G} := \lambda \overline{X^l_{\beta^l}}.\mathrm{h}\ \overline{\mathbf{V}^m}$$

$(m \geq 0)$ is called a partial binding of type α and head h (see also [SnGa89,Snyder91]), if $\mathbf{V}^i = H^i\ \overline{X^l_{\beta_l}}$ and the H^i are new variables of types $\overline{\beta^l} \rightarrow \delta^i$.

Partial bindings, where the head is a bound variable $X^j_{\beta_j}$ are called projection bindings (we write them as $\mathcal{G}^j_\alpha$) and imitation bindings (written $\mathcal{G}^h_\alpha$) otherwise.

Since we need both imitation and projection bindings for higher-order unification, we collect them in the set of general bindings for h and α ($\mathcal{AB}^h_\alpha := \{\mathcal{G}^h_\alpha\} \cup \{\mathcal{G}^j_\alpha \mid j \leq l\}$).

Def.: General Bindings

Let $\alpha := (\overline{\beta^l} \rightarrow \gamma)$ and let h be a constant or variable of type $(\overline{\delta_m} \rightarrow \gamma)$ in Σ, then

$$\mathbf{G} := \lambda \overline{X^l_{\beta^l}}.\mathsf{h}\ \overline{\mathbf{V}^m}$$

$(m \geq 0)$ is called a partial binding of type α and head h (see also [SnGa89,Snyder91]), if $\mathbf{V}^i = H^i\ \overline{X^l_{\beta_l}}$ and the H^i are new variables of types $\overline{\beta^l} \rightarrow \delta^i$.

Partial bindings, where the head is a bound variable $X^j_{\beta_j}$ are called projection bindings (we write them as $\mathcal{G}^j_\alpha$) and imitation bindings (written $\mathcal{G}^h_\alpha$) otherwise.

Since we need both imitation and projection bindings for higher-order unification, we collect them in the set of general bindings for h and α ($\mathcal{AB}^h_\alpha := \{\mathcal{G}^h_\alpha\} \cup \{\mathcal{G}^j_\alpha \mid j \leq l\}$).

Def.: Literals

- literals, e.g., $[\mathbf{A}]^{\mu}$, consist of a literal atom $\mathbf{A}$ and a polarity $\mu \in \{\mathsf{T}, \mathsf{F}\}$

Def.: Literals

- literals, e.g., $[\mathbf{A}]^\mu$, consist of a literal atom $\mathbf{A}$ and a polarity $\mu \in \{\mathsf{T}, \mathsf{F}\}$
- we distinguish between proper literals and pre-literals: the (normalised) atom of a pre-literal has a logical constant at head position, whereas this must not be the case for proper literals

Def.: Literals

- literals, e.g., $[\mathbf{A}]^{\mu}$, consist of a literal atom $\mathbf{A}$ and a polarity $\mu \in \{\mathsf{T}, \mathsf{F}\}$
- we distinguish between proper literals and pre-literals: the (normalised) atom of a pre-literal has a logical constant at head position, whereas this must not be the case for proper literals
- for instance, $[\mathbf{A} \vee \mathbf{B}]^{\mathsf{T}}$ is a pre-literal and $[\mathsf{p}_{o \rightarrow o}\ (\mathbf{A} \vee \mathbf{B})]^{\mathsf{T}}$ is a proper literal

Def.: Literals

- literals, e.g., $[\mathbf{A}]^\mu$, consist of a literal atom $\mathbf{A}$ and a polarity $\mu \in \{\mathsf{T}, \mathsf{F}\}$
- we distinguish between proper literals and pre-literals: the (normalised) atom of a pre-literal has a logical constant at head position, whereas this must not be the case for proper literals
- for instance, $[\mathbf{A} \vee \mathbf{B}]^\mathsf{T}$ is a pre-literal and $[\mathsf{p}_{o\to o}\ (\mathbf{A} \vee \mathbf{B})]^\mathsf{T}$ is a proper literal
- a literal is called flexible if its atom contains a variable at head position

Def.: Unification Constraints

- a unification problem between two terms $\mathbf{T}^1$ and $\mathbf{T}^2$ (between n terms $\mathbf{T}^1, \ldots, \mathbf{T}^n$) generated during the refutation process is called an unification constraint

Def.: Unification Constraints

- a unification problem between two terms $\mathbf{T}^1$ and $\mathbf{T}^2$ (between n terms $\mathbf{T}^1, \ldots, \mathbf{T}^{\mathsf{n}}$) generated during the refutation process is called an unification constraint
- it is represented as $[\mathbf{T}^1 \neq^? \mathbf{T}^2]$ (resp. $[\neq^? (\mathbf{T}^1, \ldots, \mathbf{T}^{\mathsf{n}})]$)

Def.: Unification Constraints

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

Def.: Unification Constraints

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

Def.: Clauses

- clauses consist of disjunctions of literals or unification constraints

Def.: Clauses

- clauses consist of disjunctions of literals or unification constraints
- the unification constraints specify conditions under which the other literals are valid

Def.: Clauses

- clauses consist of disjunctions of literals or unification constraints
- the unification constraints specify conditions under which the other literals are valid
- for instance, the clause $[\mathsf{p}_{\alpha\to\beta\to o}\ \mathbf{T}^1_\alpha\ \mathbf{T}^2_\beta]^{\mathsf{T}} \vee [\mathbf{T}^1_\alpha \neq^? \mathbf{S}^1_\alpha] \vee [\mathbf{T}^2_\beta \neq^? \mathbf{S}^2_\beta]$ can be read as: if $\mathbf{T}^1$ is unifiable with $\mathbf{S}^1$ and $\mathbf{T}^2$ with $\mathbf{S}^2$ then $(\mathsf{p}\ \mathbf{T}^1\ \mathbf{T}^2)$ holds

Def.: Clauses

- clauses consist of disjunctions of literals or unification constraints
- the unification constraints specify conditions under which the other literals are valid
- for instance, the clause $[\mathsf{p}_{\alpha\rightarrow\beta\rightarrow o}\ \mathbf{T}^1_\alpha\ \mathbf{T}^2_\beta]^{\mathsf{T}} \vee [\mathbf{T}^1_\alpha \neq^? \mathbf{S}^1_\alpha] \vee [\mathbf{T}^2_\beta \neq^? \mathbf{S}^2_\beta]$ can be read as: if $\mathbf{T}^1$ is unifiable with $\mathbf{S}^1$ and $\mathbf{T}^2$ with $\mathbf{S}^2$ then $(\mathsf{p}\ \mathbf{T}^1\ \mathbf{T}^2)$ holds
- we implicitly treat the disjunction operator $\vee$ in clauses as commutative and associative

Def.: Clauses

- clauses consist of disjunctions of literals or unification constraints
- the unification constraints specify conditions under which the other literals are valid
- for instance, the clause $[\mathsf{p}_{\alpha\to\beta\to o}\ \mathbf{T}^1_\alpha\ \mathbf{T}^2_\beta]^{\mathsf{T}} \vee [\mathbf{T}^1_\alpha \neq^? \mathbf{S}^1_\alpha] \vee [\mathbf{T}^2_\beta \neq^? \mathbf{S}^2_\beta]$ can be read as: if $\mathbf{T}^1$ is unifiable with $\mathbf{S}^1$ and $\mathbf{T}^2$ with $\mathbf{S}^2$ then $(\mathsf{p}\ \mathbf{T}^1\ \mathbf{T}^2)$ holds
- we implicitly treat the disjunction operator $\vee$ in clauses as commutative and associative
- additionally we presuppose commutativity of $\neq^?$ and implicitly identify any two α-equal constraints or literals.

Def.: Clauses

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

Def.: Clauses (contd.)

- if a clause contains at least one pre-literal we call it a pre-clause, otherwise a proper clause

Def.: Clauses (contd.)

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

Rem.: Skolemisation

- an important aspect of clause normalisation is Skolemisation

Rem.: Skolemisation

- an important aspect of clause normalisation is Skolemisation
- we employ Miller's sound adaptation of traditional first-order Skolemisation [Miller:pihol83], which associates with each Skolem function the minimum number of arguments the Skolem function has to be applied to

Rem.: Skolemisation

- an important aspect of clause normalisation is Skolemisation
- we employ Miller's sound adaptation of traditional first-order Skolemisation [Miller:pihol83], which associates with each Skolem function the minimum number of arguments the Skolem function has to be applied to
- higher-order Skolemisation becomes sound, if any Skolem function f^n only occurs in a Skolem term, i.e., a formula $\mathbf{S} = f^n\overline{\mathbf{A}^n}$, where none of the $\mathbf{A}^i$ contains a bound variable

Rem.: Skolemisation

- an important aspect of clause normalisation is Skolemisation
- we employ Miller's sound adaptation of traditional first-order Skolemisation [Miller:pihol83], which associates with each Skolem function the minimum number of arguments the Skolem function has to be applied to
- higher-order Skolemisation becomes sound, if any Skolem function f^n only occurs in a Skolem term, i.e., a formula $\mathbf{S} = f^n\overline{\mathbf{A}^n}$, 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

Rem.: Skolemisation

- an important aspect of clause normalisation is Skolemisation
- we employ Miller's sound adaptation of traditional first-order Skolemisation [Miller:pihol83], which associates with each Skolem function the minimum number of arguments the Skolem function has to be applied to
- higher-order Skolemisation becomes sound, if any Skolem function f^n only occurs in a Skolem term, i.e., a formula $\mathbf{S} = f^n\overline{\mathbf{A}^n}$, where none of the $\mathbf{A}^i$ contains a bound variable
- thus, the Skolem terms only serve as descriptions of the existential witnesses and never appear as functions proper
- without this additional restriction the calculi do not really become unsound, but one can prove an instance of the axiom of choice ([Andrews73]), which we want to treat as an optional axiom for the resolution calculi presented here

Approaches to Higher-Order Resolution: $\mathcal{R}$

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

We present and discuss Andrews' higher-order resolution calculus [Andrews71] in our uniform notation; we call this calculus $\mathcal{R}$

λ-Conversion

- Andrews' provides two rules for α-conversion and β-reduction

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

We present and discuss Andrews' higher-order resolution calculus [Andrews71] in our uniform notation; we call this calculus $\mathcal{R}$

λ-Conversion

- Andrews' provides two rules for α-conversion and β-reduction
- he does not provide a rule for η-conversion: consequently η-equality of two terms (e.g., $f_{\iota\rightarrow\iota} \doteq \lambda X_\iota . f\, X$) cannot be proven in this approach without employing the functional extensionality axiom of appropriate type

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

We present and discuss Andrews' higher-order resolution calculus [Andrews71] in our uniform notation; we call this calculus $\mathcal{R}$

λ-Conversion

- Andrews' provides two rules for α-conversion and β-reduction
- he does not provide a rule for η-conversion: consequently η-equality of two terms (e.g., $f_{\iota\rightarrow\iota} \doteq \lambda X_{\iota}.f\,X$) cannot be proven in this approach without employing the functional extensionality axiom of appropriate type
- we omit explicit rules for α- and β-convertibility and instead treat them implicitly, i.e. we assume that the presented rules operate on input and generate output in β-normal form and we automatically identify terms which differ only with respect to the names of bound variables

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

Clause Normalisation

- $\mathcal{R}$ introduces only four rules belonging to clause normalisation: negation elimination, conjunction elimination, existential elimination, and universal elimination

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

Clause Normalisation

- $\mathcal{R}$ introduces only four rules belonging to clause normalisation: negation elimination, conjunction elimination, existential elimination, and universal elimination
- as our presentation of clauses in contrast to [Andrews71] explicitly mentions the polarities of clauses and brackets the literal atoms we need additional structural rules, e.g., the rule $\vee^T$

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

Clause Normalisation

- $\mathcal{R}$ introduces only four rules belonging to clause normalisation: negation elimination, conjunction elimination, existential elimination, and universal elimination
- as our presentation of clauses in contrast to [Andrews71] explicitly mentions the polarities of clauses and brackets the literal atoms we need additional structural rules, e.g., the rule $\vee^T$
- negation elimination: $\dfrac{\mathbf{C} \vee [\neg \mathbf{A}]^T}{\mathbf{C} \vee [\mathbf{A}]^F}\ \neg^T \qquad \dfrac{\mathbf{C} \vee [\neg \mathbf{A}]^F}{\mathbf{C} \vee [\mathbf{A}]^T}\ \neg^F$

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

Clause Normalisation

- $\mathcal{R}$ introduces only four rules belonging to clause normalisation: negation elimination, conjunction elimination, existential elimination, and universal elimination
- as our presentation of clauses in contrast to [Andrews71] explicitly mentions the polarities of clauses and brackets the literal atoms we need additional structural rules, e.g., the rule $\vee^T$
- negation elimination: $$\frac{\mathbf{C} \vee [\neg \mathbf{A}]^T}{\mathbf{C} \vee [\mathbf{A}]^F}\ \neg^T \qquad \frac{\mathbf{C} \vee [\neg \mathbf{A}]^F}{\mathbf{C} \vee [\mathbf{A}]^T}\ \neg^F$$
- conjunction/disjunction elimination:

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

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

Clause Normalisation (contd.)

- existential/universal elimination:

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

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

Clause Normalisation (contd.)

- existential/universal elimination:

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

X_{α} is a new free variable and s_{α} is a new Skolem term

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

Clause Normalisation (contd.)

- existential/universal elimination:

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

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

- additionally Andrews presents rules addressing commutativity and associativity of the $\vee$-operator connecting the clauses literals; we have already mentioned the implicit treatment of these aspects here

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

Clause Normalisation (contd.)

- existential/universal elimination:

$$\frac{\mathbf{C} \lor [\Pi^{\alpha}\mathbf{A}]^{\mathsf{T}}}{\mathbf{C} \lor [\mathbf{A}\ \mathsf{X}_{\alpha}]^{\mathsf{T}}}\ \Pi^{\mathsf{T}} \qquad \frac{\mathbf{C} \lor [\Pi^{\alpha}\mathbf{A}]^{\mathsf{F}}}{\mathbf{C} \lor [\mathbf{A}\ \mathsf{s}_{\alpha}]^{\mathsf{F}}}\ \Pi^{\mathsf{F}}$$

  X_{α} is a new free variable and s_{α} is a new Skolem term

- additionally Andrews presents rules addressing commutativity and associativity of the $\lor$-operator connecting the clauses literals; we have already mentioned the implicit treatment of these aspects here
- we refer with $\mathsf{Cnf}(\mathbf{A})$ to the set of clauses obtained from formula $\mathbf{A}$ by exhaustive clause normalisation

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

Resolution & Factorisation

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

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

Resolution & Factorisation

- Instead of a resolution and a factorisation rule — which work in connection with unification — Andrews presents a simplification and a cut rule. The cut rule is only applicable to clauses with two complementary literals which have identical atoms. Similarly Sim is defined only for clauses with two identical literals. In order to generate identical literal atoms during the refutation process these two rules have to be combined with the substitution rule Sub presented below.
- Simplification: $$\frac{[\mathbf{A}]^{\mu} \vee [\mathbf{A}]^{\mu} \vee \mathbf{C}}{[\mathbf{A}]^{\mu} \vee \mathbf{C}} \text{ Sim}$$

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

Resolution & Factorisation

- Instead of a resolution and a factorisation rule — which work in connection with unification — Andrews presents a simplification and a cut rule. The cut rule is only applicable to clauses with two complementary literals which have identical atoms. Similarly Sim is defined only for clauses with two identical literals. In order to generate identical literal atoms during the refutation process these two rules have to be combined with the substitution rule Sub presented below.
- Simplification: $$\frac{[\mathbf{A}]^{\mu} \vee [\mathbf{A}]^{\mu} \vee \mathbf{C}}{[\mathbf{A}]^{\mu} \vee \mathbf{C}} \text{ Sim}$$
- Cut: $$\frac{[\mathbf{A}]^{\mu} \vee \mathbf{C} \quad [\mathbf{A}]^{\nu} \vee \mathbf{D}}{\mathbf{C} \vee \mathbf{D}} \text{ Cut}$$

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

Unification & Primitive Substitution

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

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

Unification & Primitive Substitution

- As higher-order unification was still an open problem in 1971 calculus $\mathcal{R}$ employs the British museum method instead, i.e. it provides a substitution rule that allows to blindly instantiate free variables by arbitrary terms. As the instantiated terms may contain logical constants, instantiation of variables in proper clauses may lead to pre-clauses, which must be normalised again with the clause normalisation rules.
- Substitution of arbitrary terms: $$\frac{\mathcal{C}}{\mathcal{C}_{[\mathbf{T}_\alpha/\mathsf{X}_\alpha]}}\ \mathrm{Sub}$$

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

Unification & Primitive Substitution

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

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

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

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

Extensionality Treatment

- Calculus $\mathcal{R}$ does not provide rules addressing the functional and/or Boolean extensionality principles.

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

Extensionality Treatment

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

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

Extensionality Treatment

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

$$\mathbf{EXT}^{\doteq}_{\alpha\to\beta}: \quad \forall F_{\alpha\to\beta}. \forall G_{\alpha\to\beta}. (\forall X_\beta. F\,X \doteq G\,X) \Rightarrow F \doteq G$$

$$\mathbf{EXT}^{\doteq}_{o}: \quad \forall A_o. \forall B_o. (A \Leftrightarrow B) \Rightarrow A \doteq^{o} B$$

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

Extensionality Treatment (contd.)

- The extensionality clauses derived from the extensionality axioms have the following form (note the many free variables, especially at literal head position, that are introduced into the search space – they heavily increase the amount of blind search in any attempt to automate the calculus):

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

Extensionality Treatment (contd.)

- The extensionality clauses derived from the extensionality axioms have the following form (note the many free variables, especially at literal head position, that are introduced into the search space – they heavily increase the amount of blind search in any attempt to automate the calculus):

$$\mathcal{E}_{\mathbf{1}}^{\alpha\to\beta} : [\mathsf{p}\ (\mathbf{F}\ \mathsf{s})]^{\mathsf{T}} \vee [\mathbf{Q}\ \mathbf{F}]^{\mathsf{F}} \vee [\mathbf{Q}\ \mathbf{G}]^{\mathsf{T}}$$
$$\mathcal{E}_{\mathbf{2}}^{\alpha\to\beta} : [\mathsf{p}\ (\mathbf{G}\ \mathsf{s})]^{\mathsf{F}} \vee [\mathbf{Q}\ \mathbf{F}]^{\mathsf{F}} \vee [\mathbf{Q}\ \mathbf{G}]^{\mathsf{T}}$$

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

Extensionality Treatment (contd.)

- The extensionality clauses derived from the extensionality axioms have the following form (note the many free variables, especially at literal head position, that are introduced into the search space – they heavily increase the amount of blind search in any attempt to automate the calculus):

$$\mathcal{E}_{\mathbf{1}}^{\alpha\to\beta} : [\mathsf{p}\ (\mathbf{F}\ \mathsf{s})]^{\mathsf{T}} \lor [\mathbf{Q}\ \mathbf{F}]^{\mathsf{F}} \lor [\mathbf{Q}\ \mathbf{G}]^{\mathsf{T}}$$
$$\mathcal{E}_{\mathbf{2}}^{\alpha\to\beta} : [\mathsf{p}\ (\mathbf{G}\ \mathsf{s})]^{\mathsf{F}} \lor [\mathbf{Q}\ \mathbf{F}]^{\mathsf{F}} \lor [\mathbf{Q}\ \mathbf{G}]^{\mathsf{T}}$$

$$\mathcal{E}_{\mathbf{1}}^{\mathbf{o}} : [\mathbf{A}]^{\mathsf{F}} \lor [\mathbf{B}]^{\mathsf{F}} \lor [\mathbf{P}\ \mathbf{A}]^{\mathsf{F}} \lor [\mathbf{P}\ \mathbf{B}]^{\mathsf{T}}$$
$$\mathcal{E}_{\mathbf{2}}^{\mathbf{o}} : [\mathbf{A}]^{\mathsf{T}} \lor [\mathbf{B}]^{\mathsf{T}} \lor [\mathbf{P}\ \mathbf{A}]^{\mathsf{F}} \lor [\mathbf{P}\ \mathbf{B}]^{\mathsf{T}}$$

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

Extensionality Treatment (contd.)

- The extensionality clauses derived from the extensionality axioms have the following form (note the many free variables, especially at literal head position, that are introduced into the search space – they heavily increase the amount of blind search in any attempt to automate the calculus):

$$\mathcal{E}_{\mathbf{1}}^{\alpha\to\beta} : [\mathsf{p}\ (\mathbf{F}\ \mathsf{s})]^{\mathsf{T}} \vee [\mathbf{Q}\ \mathbf{F}]^{\mathsf{F}} \vee [\mathbf{Q}\ \mathbf{G}]^{\mathsf{T}}$$
$$\mathcal{E}_{\mathbf{2}}^{\alpha\to\beta} : [\mathsf{p}\ (\mathbf{G}\ \mathsf{s})]^{\mathsf{F}} \vee [\mathbf{Q}\ \mathbf{F}]^{\mathsf{F}} \vee [\mathbf{Q}\ \mathbf{G}]^{\mathsf{T}}$$

$$\mathcal{E}_{\mathbf{1}}^{\mathbf{o}} : [\mathbf{A}]^{\mathsf{F}} \vee [\mathbf{B}]^{\mathsf{F}} \vee [\mathbf{P}\ \mathbf{A}]^{\mathsf{F}} \vee [\mathbf{P}\ \mathbf{B}]^{\mathsf{T}}$$
$$\mathcal{E}_{\mathbf{2}}^{\mathbf{o}} : [\mathbf{A}]^{\mathsf{T}} \vee [\mathbf{B}]^{\mathsf{T}} \vee [\mathbf{P}\ \mathbf{A}]^{\mathsf{F}} \vee [\mathbf{P}\ \mathbf{B}]^{\mathsf{T}}$$

$\mathsf{p}_{\beta\to o}$, s_{α} are Skolem terms and A_o, B_o, $\mathsf{P}_{o\to o}$, $\mathsf{Q}_{(\alpha\to\beta)\to o}$ are new free variables.

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

Proof Search

- initially the proof problem is negated and normalised

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

Proof Search

- initially the proof problem is negated and normalised
- proof search then starts with the normalised clauses and applies the cut and simplification rule in close connection with the substitution rule

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

Proof Search

- initially the proof problem is negated and normalised
- proof search then starts with the normalised clauses and applies the cut and simplification rule in close connection with the substitution rule
- intermediate applications of the clause normalisation rules may be needed to normalise temporarily generated pre-clauses

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

Proof Search

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

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

Proof Search

- initially the proof problem is negated and normalised
- proof search then starts with the normalised clauses and applies the cut and simplification rule in close connection with the substitution rule
- intermediate applications of the clause normalisation rules may be needed to normalise temporarily generated pre-clauses
- the extensionality treatment in $\mathcal{R}$ simply assumes to add at the beginning of the refutation process the above clauses obtained from the extensionality axioms
- the proof search can be graphically illustrated as follows:

ext. axioms	proof search & blind variable instantiation

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

Completeness

- [Andrews71] gives a completeness proof for calculus $\mathcal{R}$ with respect to the semantical notion of V-complexes (corresponds to our weakest model class $\mathfrak{M}_\beta(\Sigma)$)

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

Completeness

- [Andrews71] gives a completeness proof for calculus $\mathcal{R}$ with respect to the semantical notion of V-complexes (corresponds to our weakest model class $\mathfrak{M}_\beta(\Sigma)$)
- as the extensionality principles are not valid in this rather weak semantical structures, the extensionality axioms are not needed in this completeness proof

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

Completeness

- [Andrews71] gives a completeness proof for calculus $\mathcal{R}$ with respect to the semantical notion of V-complexes (corresponds to our weakest model class $\mathfrak{M}_\beta(\Sigma)$)
- as the extensionality principles are not valid in this rather weak semantical structures, the extensionality axioms are not needed in this completeness proof
- Theorem: (V-completeness of $\mathcal{R}$) The calculus $\mathcal{R}$ is (sound and) complete with respect to the notion of V-complexes.

  Proof: [Andrews71].

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

Henkin Completeness

- We can also prove Henkin completeness of calculus $\mathcal{R}$.

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

Henkin Completeness

- We can also prove Henkin completeness of calculus $\mathcal{R}$.
- Theorem: (Henkin completeness of $\mathcal{R}$) The calculus $\mathcal{R}$ is (sound and) complete with respect to Henkin semantics provided that the infinitely many extensionality axioms are given.

  Proof: exercise

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

Henkin Completeness

- We can also prove Henkin completeness of calculus $\mathcal{R}$.
- Theorem: (Henkin completeness of $\mathcal{R}$) The calculus $\mathcal{R}$ is (sound and) complete with respect to Henkin semantics provided that the infinitely many extensionality axioms are given.

  Proof: exercise

Example Proofs

Exercise: How are the following theorems proved in calculus $\mathcal{R}$?

- Leibniz equality and η-equality:

$$\mathsf{f}_{\iota\to\iota} \doteq \lambda \mathsf{X}_\iota\mathbf{.}\mathsf{f}\ \mathsf{X}$$

Example Proofs

Exercise: How are the following theorems proved in calculus $\mathcal{R}$?

- Leibniz equality and η-equality:

$$\mathsf{f}_{\iota\rightarrow\iota} \doteq \lambda \mathsf{X}_\iota.\mathsf{f}\ \mathsf{X}$$

- The set of all red balls equals the set of all balls that are red: $\{X|\text{red } X \wedge \text{ball } X\} = \{X|\text{ball } X \wedge \text{red } X\}$. This problem can be encoded as

$$(\lambda X_\iota.\text{red } X \wedge \text{ball } X) = (\lambda X_\iota.\text{ball } X \wedge \text{red } X)$$

Example Proofs

Exercise: How are the following theorems proved in calculus $\mathcal{R}$?

- All unary logical operators $O_{o \to o}$ which map the propositions a and b to $\top$ consequently also map $a \wedge b$ to $\top$:

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

Example Proofs

Exercise: How are the following theorems proved in calculus $\mathcal{R}$?

- In Henkin semantics the domain $\mathcal{D}_o$ of all Booleans contains exactly the truth values $\bot$ and $\top$. Consequently the domain of all mappings from Booleans to Booleans contains exactly contains in each Henkin model at most four elements. And because of the requirement, that the function domains in Henkin models must be rich enough such that every term has a denotation, it follows that $\mathcal{D}_{o\to o}$ contains exactly the pairwise distinct denotations of the following four terms: $\lambda X_o . X_o$, $\lambda X_o . \neg X_o$, $\lambda X_o . \bot$, and $\lambda X_o . \top$. This theorem can be formulated as follows (where $f_{o\to o}$ is a constant):

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

Approaches to Higher-Order Resolution: $\mathcal{CR}$

Huet's Constrained Resolution $\mathcal{CR}$

We transform Huet's constrained resolution approach [Huet72,Huet73] in our uniform notation. The calculus here is the unsorted fragment of the variant of Huet's approach as presented in [Kohlhase94]. In the remainder of this paper we refer to this calculus with $\mathcal{CR}$.

λ-Conversion

- Calculus $\mathcal{CR}$ assumes that terms, literals, and clauses are implicitly reduced to β-normal form.

Huet's Constrained Resolution $\mathcal{CR}$

We transform Huet's constrained resolution approach [Huet72,Huet73] in our uniform notation. The calculus here is the unsorted fragment of the variant of Huet's approach as presented in [Kohlhase94]. In the remainder of this paper we refer to this calculus with $\mathcal{CR}$.

λ-Conversion

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

Huet's Constrained Resolution $\mathcal{CR}$

Clause Normalisation

- [Huet72] does not explicitly present clause normalisation rules but assumes that they are given. Here we employ the rules $\neg^T$, $\neg^F$, $\vee^T$, $\vee^F_l$, $\vee^F_r$, Π^T, and Π^F as already defined for calculus $\mathcal{R}$ before.

Huet's Constrained Resolution $\mathcal{CR}$

Clause Normalisation

- [Huet72] does not explicitly present clause normalisation rules but assumes that they are given. Here we employ the rules $\neg^T$, $\neg^F$, $\vee^T$, $\vee^F_l$, $\vee^F_r$, Π^T, and Π^F as already defined for calculus $\mathcal{R}$ before.

Huet's Constrained Resolution $\mathcal{CR}$

Resolution & Factorisation

- As first-order unification is decidable and unitary it can be employed as a strong filter in first-order resolution [Robinson65].

Huet's Constrained Resolution $\mathcal{CR}$

Resolution & Factorisation

- As first-order unification is decidable and unitary it can be employed as a strong filter in first-order resolution [Robinson65].
- Unfortunately higher-order unification is not decidable (cf. [Lucchesi72,Huet73,Goldfarb81]) and thus it can not be applied in the sense of a terminating side computation in higher-order theorem proving.

Huet's Constrained Resolution $\mathcal{CR}$

Resolution & Factorisation

- As first-order unification is decidable and unitary it can be employed as a strong filter in first-order resolution [Robinson65].
- Unfortunately higher-order unification is not decidable (cf. [Lucchesi72,Huet73,Goldfarb81]) and thus it can not be applied in the sense of a terminating side computation in higher-order theorem proving.
- Huet therefore suggests in [Huet72,Huet73] to delay the unification process and to explicitly encode unification problems occurring during the refutation search as unification constraints.

Huet's Constrained Resolution $\mathcal{CR}$

Resolution & Factorisation (contd.)

- In his original approach Huet presented a hyper-resolution rule which simultaneously resolves on the resolution literals $\mathbf{A}^1, \ldots \mathbf{A}^{\mathsf{n}}$ $(1 \leq \mathsf{n})$ and $\mathbf{B}^1, \ldots \mathbf{B}^{\mathsf{m}}$ $(1 \leq \mathsf{m})$ of two given clauses and adds the unification constraint $[\neq^? (\mathbf{A}^1, \ldots \mathbf{A}^{\mathsf{n}}, \mathbf{B}^1, \ldots \mathbf{B}^{\mathsf{m}})]$ to the resolvent:

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

(where $\mu \neq \nu$).

Huet's Constrained Resolution $\mathcal{CR}$

Resolution & Factorisation (contd.)

- In order to ease the comparison with the two other approaches discussed in this paper we instead employ a resolution rule Res and a factorisation rule Fac.

Huet's Constrained Resolution $\mathcal{CR}$

Resolution & Factorisation (contd.)

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

Huet's Constrained Resolution $\mathcal{CR}$

Resolution & Factorisation (contd.)

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

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

(where $\mu \neq \nu$).

Huet's Constrained Resolution $\mathcal{CR}$

Resolution & Factorisation (contd.)

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

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

(where $\mu \neq \nu$).

- Constrained factorisation:

$$\frac{[\mathbf{A}]^{\mu} \vee [\mathbf{B}]^{\mu} \vee \mathbf{C}}{[\mathbf{A}]^{\mu} \vee \mathbf{C} \vee [\mathbf{A} \neq^{?} \mathbf{B}]^{\mathsf{F}}} \text{ Fac}$$

Huet's Constrained Resolution $\mathcal{CR}$

Resolution & Factorisation (contd.)

- One can easily prove by induction on $n + m$ that each proof step applying rule Hres can be replaced by a corresponding derivation employing Res and Fac.

Huet's Constrained Resolution $\mathcal{CR}$

Resolution & Factorisation (contd.)

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

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting

- [Huet75] introduces higher-order unification and higher-order pre-unification and shows that higher-order pre-unification is sufficient to verify the soundness of a refutation in which the occurring unification problems have been delayed until the end.

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting

- [Huet75] introduces higher-order unification and higher-order pre-unification and shows that higher-order pre-unification is sufficient to verify the soundness of a refutation in which the occurring unification problems have been delayed until the end.
- The higher-order pre-unification rules presented here are discussed in detail in [Benzmüller-PhD-99]. They furthermore closely reflect the rules as presented in [SnyderGallier89].

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting

- [Huet75] introduces higher-order unification and higher-order pre-unification and shows that higher-order pre-unification is sufficient to verify the soundness of a refutation in which the occurring unification problems have been delayed until the end.
- The higher-order pre-unification rules presented here are discussed in detail in [Benzmüller-PhD-99]. They furthermore closely reflect the rules as presented in [SnyderGallier89].
- Elimination of trivial pairs:

$$\frac{\mathbf{C} \vee [\mathbf{A} \neq^{?} \mathbf{A}]}{\mathbf{C}}\ \text{Triv}$$

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting

- [Huet75] introduces higher-order unification and higher-order pre-unification and shows that higher-order pre-unification is sufficient to verify the soundness of a refutation in which the occurring unification problems have been delayed until the end.
- The higher-order pre-unification rules presented here are discussed in detail in [Benzmüller-PhD-99]. They furthermore closely reflect the rules as presented in [SnyderGallier89].
- Elimination of trivial pairs: $$\frac{\mathbf{C} \vee [\mathbf{A} \neq^? \mathbf{A}]}{\mathbf{C}}\ \mathrm{Triv}$$
- Decomposition $$\frac{\mathsf{C} \vee [\mathsf{h}\overline{\mathbf{U}^n} \neq^? \mathsf{h}\overline{\mathbf{V}^n}]}{\mathsf{C} \vee [\mathbf{U}^1 \neq^? \mathbf{V}^1] \vee \ldots \vee [\mathbf{U}^n \neq^? \mathbf{V}^n]}\ \mathrm{Dec}$$

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- Elimination of λ-binders: (weak functional extensionality)

$$\frac{\mathbf{C} \vee [\mathbf{M}_{\alpha\rightarrow\beta} \neq^{?} \mathbf{N}_{\alpha\rightarrow\beta}]}{\mathbf{C} \vee [\mathbf{M}\ \mathsf{s}_{\alpha} \neq^{?} \mathbf{N}\ \mathsf{s}_{\alpha}]}\ \text{Func}$$

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- Elimination of λ-binders: (weak functional extensionality)

$$\frac{\mathbf{C} \vee [\mathbf{M}_{\alpha\to\beta} \neq^? \mathbf{N}_{\alpha\to\beta}]}{\mathbf{C} \vee [\mathbf{M}\ \mathsf{s}_\alpha \neq^? \mathbf{N}\ \mathsf{s}_\alpha]}\ \text{Func}$$

s_α is a new Skolem term.

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- Elimination of λ-binders: (weak functional extensionality)

$$\frac{\mathbf{C} \vee [\mathbf{M}_{\alpha\to\beta} \neq^? \mathbf{N}_{\alpha\to\beta}]}{\mathbf{C} \vee [\mathbf{M}\, \mathsf{s}_\alpha \neq^? \mathbf{N}\, \mathsf{s}_\alpha]}\ \text{Func}$$

s_α is a new Skolem term.

- Imitation of rigid heads:

$$\frac{\mathbf{C} \vee [\mathsf{F}_\gamma\, \overline{\mathbf{U}^{\mathsf{n}}} \neq^? \mathsf{h}\, \overline{\mathbf{V}^{\mathsf{m}}}] \quad \mathbf{G} \in \mathcal{AB}^{\mathsf{h}}_\gamma}{\mathbf{C} \vee [\mathsf{F} \neq^? \mathbf{G}] \vee [\mathsf{F}\, \overline{\mathbf{U}^{\mathsf{n}}} \neq^? \mathsf{h}\, \overline{\mathbf{V}^{\mathsf{m}}}]}\ \text{FlexRigid}$$

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- Elimination of λ-binders: (weak functional extensionality)

$$\frac{\mathbf{C} \vee [\mathbf{M}_{\alpha\to\beta} \neq^? \mathbf{N}_{\alpha\to\beta}]}{\mathbf{C} \vee [\mathbf{M}\ \mathsf{s}_\alpha \neq^? \mathbf{N}\ \mathsf{s}_\alpha]}\ \text{Func}$$

s_α is a new Skolem term.

- Imitation of rigid heads:

$$\frac{\mathbf{C} \vee [\mathsf{F}_\gamma\ \overline{\mathbf{U}^{\mathsf{n}}} \neq^? \mathsf{h}\ \overline{\mathbf{V}^{\mathsf{m}}}] \quad \mathbf{G} \in \mathcal{AB}^{\mathsf{h}}_\gamma}{\mathbf{C} \vee [\mathsf{F} \neq^? \mathbf{G}] \vee [\mathsf{F}\ \overline{\mathbf{U}^{\mathsf{n}}} \neq^? \mathsf{h}\ \overline{\mathbf{V}^{\mathsf{m}}}]}\ \text{FlexRigid}$$

$\mathcal{AB}^{\mathsf{h}}_\gamma$ is the set of general bindings of type γ for head h.

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- Huet points to the usefulness of eager unification to filter out clauses with non-unifiable unification constraints or to back-propagate the solutions of easily solvable constraints (e.g., in case of first-order unification problems occurring during the proof search): many of the higher-order unification problems occurring in practice are decidable and have only finitely many solutions.

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- Huet points to the usefulness of eager unification to filter out clauses with non-unifiable unification constraints or to back-propagate the solutions of easily solvable constraints (e.g., in case of first-order unification problems occurring during the proof search): many of the higher-order unification problems occurring in practice are decidable and have only finitely many solutions.
- Hence, even though higher-order unification is generally not decidable it is sensible in practice to apply the unification algorithm with a particular resource, such that only those unification problems which may have further solutions beyond this bound need to be delayed.

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- In our presentation of calculus $\mathcal{CR}$ we explicitly address the aspect of eager unification and substitution by rule Subst. This rule back-propagates eagerly computed unifiers to the literal part of a clause.

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- In our presentation of calculus $\mathcal{CR}$ we explicitly address the aspect of eager unification and substitution by rule Subst. This rule back-propagates eagerly computed unifiers to the literal part of a clause.
- Eager unification & substitution:

$$\frac{\mathbf{C} \lor [\mathsf{X} \neq^{?} \mathbf{A}] \quad \mathsf{X} \notin \mathsf{free}(\mathbf{A})}{\mathbf{C}_{[\mathbf{A}/\mathsf{X}]}} \ \mathsf{Subst}$$

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- The literal heads of our clauses may consist of set variables and it may be necessary to instantiate them with terms introducing new logical constant at head position in order to find a refutation.

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- The literal heads of our clauses may consist of set variables and it may be necessary to instantiate them with terms introducing new logical constant at head position in order to find a refutation.
- Unfortunately not all appropriate instantiations can be computed with the calculus rules presented so far.

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

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

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- Instantiate
  set variables:

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- Instantiate set variables:

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mathsf{T}} \vee \mathbf{C}}{[\mathsf{Q}]^{\mathsf{T}} \vee [\mathsf{R}]^{\mathsf{T}} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A}\ \neq^{?} (\mathsf{Q}_{o} \vee \mathsf{R}_{o})]}\ \mathsf{S}^{\mathsf{T}}_{\vee}$$

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- Instantiate set variables:

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mathsf{T}} \vee \mathbf{C}}{[\mathsf{Q}]^{\mathsf{T}} \vee [\mathsf{R}]^{\mathsf{T}} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A}\ \neq^{?} (\mathsf{Q}_{\mathsf{o}} \vee \mathsf{R}_{\mathsf{o}})]}\ \mathsf{S}^{\mathsf{T}}_{\vee}$$

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mathsf{F}} \vee \mathbf{C}}{[\mathsf{Q}]^{\mathsf{F}} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A}\ \neq^{?} (\mathsf{Q}_{\mathsf{o}} \vee \mathsf{R}_{\mathsf{o}})] \quad [\mathsf{R}]^{\mathsf{F}} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A}\ \neq^{?} (\mathsf{Q}_{\mathsf{o}} \vee \mathsf{R}_{\mathsf{o}})]}\ \mathsf{S}^{\mathsf{F}}_{\vee}$$

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mu} \vee \mathbf{C}}{[\mathsf{Q}]^{\nu} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A}\ \neq^{?} \neg \mathsf{Q}_{\mathsf{o}}]}\ \mathsf{S}^{\mathsf{TF}}_{\neg}$$

(where $\mu \neq \nu$)

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- Instantiate set variables:

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mathsf{T}} \vee \mathbf{C}}{[\mathsf{Q}]^{\mathsf{T}} \vee [\mathsf{R}]^{\mathsf{T}} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A}\ \ \neq^{?}\ (\mathsf{Q}_{\mathsf{o}} \vee \mathsf{R}_{\mathsf{o}})]}\ \mathsf{S}_{\vee}^{\mathsf{T}}$$

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mu} \vee \mathbf{C}}{[\mathsf{Q}]^{\nu} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A}\ \ \neq^{?}\ \neg\mathsf{Q}_{\mathsf{o}}]}\ \mathsf{S}_{\neg}^{\mathsf{TF}}$$

(where $\mu \neq \nu$)

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mathsf{F}} \vee \mathbf{C}}{[\mathsf{Q}]^{\mathsf{F}} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A}\ \ \neq^{?}\ (\mathsf{Q}_{\mathsf{o}} \vee \mathsf{R}_{\mathsf{o}})] \qquad [\mathsf{R}]^{\mathsf{F}} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A}\ \ \neq^{?}\ (\mathsf{Q}_{\mathsf{o}} \vee \mathsf{R}_{\mathsf{o}})]}\ \mathsf{S}_{\vee}^{\mathsf{F}}$$

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

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- Instantiate set variables:

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mathsf{T}} \lor \mathbf{C}}{[\mathsf{Q}]^{\mathsf{T}} \lor [\mathsf{R}]^{\mathsf{T}} \lor \mathbf{C} \lor [\mathsf{P}\ \mathbf{A}\ \neq^{?} (\mathsf{Q}_{\mathsf{o}} \lor \mathsf{R}_{\mathsf{o}})]}\ \mathsf{S}_{\lor}^{\mathsf{T}}$$

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mu} \lor \mathbf{C}}{[\mathsf{Q}]^{\nu} \lor \mathbf{C} \lor [\mathsf{P}\ \mathbf{A}\ \neq^{?} \neg\mathsf{Q}_{\mathsf{o}}]}\ \mathsf{S}_{\neg}^{\mathsf{TF}}$$

(where $\mu \neq \nu$)

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mathsf{F}} \lor \mathbf{C}}{\begin{array}{c}[\mathsf{Q}]^{\mathsf{F}} \lor \mathbf{C} \lor [\mathsf{P}\ \mathbf{A}\ \neq^{?} (\mathsf{Q}_{\mathsf{o}} \lor \mathsf{R}_{\mathsf{o}})]\\ [\mathsf{R}]^{\mathsf{F}} \lor \mathbf{C} \lor [\mathsf{P}\ \mathbf{A}\ \neq^{?} (\mathsf{Q}_{\mathsf{o}} \lor \mathsf{R}_{\mathsf{o}})]\end{array}}\ \mathsf{S}_{\lor}^{\mathsf{F}}$$

$$\frac{[\mathsf{P}\ \mathbf{A}_{\alpha\to\mathsf{o}}]^{\mathsf{T}} \lor \mathbf{C}}{[\mathsf{M}_{\alpha\to\mathsf{o}}\ \mathsf{Z}]^{\mathsf{T}} \lor \mathbf{C} \lor [\mathsf{P}\ \mathbf{A}\ \neq^{?} \Pi^{\alpha}\mathsf{M}]}\ \mathsf{S}_{\Pi}^{\mathsf{T}}$$

$$\frac{[\mathsf{P}\ \mathbf{A}_{\alpha\to\mathsf{o}}]^{\mathsf{F}} \lor \mathbf{C}}{[\mathsf{M}_{\alpha\to\mathsf{o}}\ \mathsf{s}]^{\mathsf{F}} \lor \mathbf{C} \lor [\mathsf{P}\ \mathbf{A}\ \neq^{?} \Pi^{\alpha}\mathsf{M}]}\ \mathsf{S}_{\Pi}^{\mathsf{F}}$$

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- Instantiate set variables:

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mathsf{T}} \vee \mathbf{C}}{[\mathsf{Q}]^{\mathsf{T}} \vee [\mathsf{R}]^{\mathsf{T}} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A} \ \neq^? (\mathsf{Q}_o \vee \mathsf{R}_o)]}\ \mathsf{S}^{\mathsf{T}}_{\vee}$$

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mathsf{F}} \vee \mathbf{C}}{[\mathsf{Q}]^{\mathsf{F}} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A} \ \neq^? (\mathsf{Q}_o \vee \mathsf{R}_o)] \quad [\mathsf{R}]^{\mathsf{F}} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A} \ \neq^? (\mathsf{Q}_o \vee \mathsf{R}_o)]}\ \mathsf{S}^{\mathsf{F}}_{\vee}$$

$$\frac{[\mathsf{P}\ \mathbf{A}]^{\mu} \vee \mathbf{C}}{[\mathsf{Q}]^{\nu} \vee \mathbf{C} \vee [\mathsf{P}\ \mathbf{A} \ \neq^? \neg \mathsf{Q}_o]}\ \mathsf{S}^{\mathsf{TF}}_{\neg}$$

(where $\mu \neq \nu$)

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

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

- $\mathsf{S}^{\mathsf{T}}_{\Pi}$ and $\mathsf{S}^{\mathsf{F}}_{\Pi}$ are infinitely branching as they are parameterised over type α. $\mathsf{Q}_o, \mathsf{R}_o, \mathsf{M}_{\alpha \to o}, \mathsf{Z}_\alpha$ are new variables and s_α is a new Skolem constant.

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- A theorem which is not refutable in $\mathcal{CR}$ if the splitting rules are not available is $\exists A_o.A$:

Huet's Constrained Resolution $\mathcal{CR}$

Unification & Splitting (contd.)

- A theorem which is not refutable in $\mathcal{CR}$ if the splitting rules are not available is $\exists A_o.A$:
- After negation this statement normalises to clause $\mathcal{C}_1 : [A]^F$, such that none but the splitting rules are applicable. With the help of rule $S_{\neg}^{TF}$ and eager unification, however, we can derive $\mathcal{C}_2 : [A']^T$ which is then successfully resolvable against $\mathcal{C}_1$.

Huet's Constrained Resolution $\mathcal{CR}$

Extensionality Treatment

- On the one hand η-convertibility is built-in in higher-order unification, such that calculus $\mathcal{CR}$ already supports functional extensionality reasoning to a certain extend.

Huet's Constrained Resolution $\mathcal{CR}$

Extensionality Treatment

- On the one hand η-convertibility is built-in in higher-order unification, such that calculus $\mathcal{CR}$ already supports functional extensionality reasoning to a certain extend.
- On the other hand $\mathcal{CR}$ nevertheless fails to address full extensionality as it does not realise the required subtle interplay between the functional and Boolean extensionality principles.

Huet's Constrained Resolution $\mathcal{CR}$

Extensionality Treatment

- On the one hand η-convertibility is built-in in higher-order unification, such that calculus $\mathcal{CR}$ already supports functional extensionality reasoning to a certain extend.
- On the other hand $\mathcal{CR}$ nevertheless fails to address full extensionality as it does not realise the required subtle interplay between the functional and Boolean extensionality principles.
- Without employing additional (Boolean and functional!) extensionality axioms $\mathcal{CR}$ is, e.g., not able to prove the rather simple examples presented before.

Huet's Constrained Resolution $\mathcal{CR}$

Proof Search

- Initially the proof problem is negated and normalised. The main proof search then operates on the generated clauses by applying the resolution, factorisation, and splitting rules.

Huet's Constrained Resolution $\mathcal{CR}$

Proof Search

- Initially the proof problem is negated and normalised. The main proof search then operates on the generated clauses by applying the resolution, factorisation, and splitting rules.
- Despite the possibility of eager unification $\mathcal{CR}$ generally foresees to delay the higher-order unification process in order to overcome the undecidability problem.

Huet's Constrained Resolution $\mathcal{CR}$

Proof Search

- Initially the proof problem is negated and normalised. The main proof search then operates on the generated clauses by applying the resolution, factorisation, and splitting rules.
- Despite the possibility of eager unification $\mathcal{CR}$ generally foresees to delay the higher-order unification process in order to overcome the undecidability problem.
- When deriving a potentially empty clause (no normal literals), $\mathcal{CR}$ then tests whether the accumulated unification constraints justifying this particular refutation are solvable.

Huet's Constrained Resolution $\mathcal{CR}$

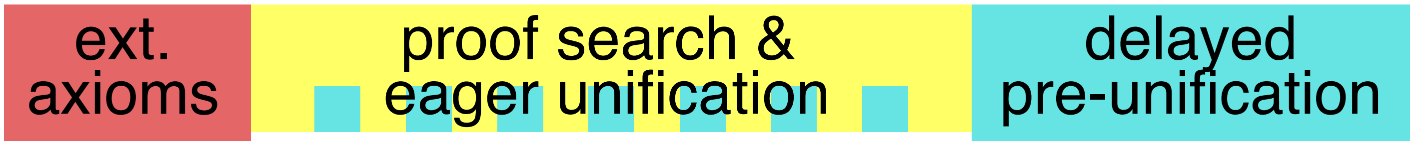
Proof Search (contd.)

- Like $\mathcal{R}$, the extensionality treatment of $\mathcal{CR}$ requires to add infinitely many extensionality axioms to the search space.

Huet's Constrained Resolution $\mathcal{CR}$

Proof Search (contd.)

- Like $\mathcal{R}$, the extensionality treatment of $\mathcal{CR}$ requires to add infinitely many extensionality axioms to the search space.
- The following figure graphically illustrates the main ideas of the proof search in $\mathcal{CR}$.



Huet's Constrained Resolution $\mathcal{CR}$

Completeness Results

- [Huet72,Huet73] analyses completeness of $\mathcal{CR}$ formally only with respect to Andrews V-complexes, i.e. Huet verifies that the set of non-refutable sentences in $\mathcal{CR}$ is an abstract consistency class for V-complexes.

Huet's Constrained Resolution $\mathcal{CR}$

Completeness Results

- [Huet72,Huet73] analyses completeness of $\mathcal{CR}$ formally only with respect to Andrews V-complexes, i.e. Huet verifies that the set of non-refutable sentences in $\mathcal{CR}$ is an abstract consistency class for V-complexes.
- Theorem (V-completeness of $\mathcal{CR}$): The calculus $\mathcal{CR}$ is complete with respect to the notion of V-complexes.

  Proof: [Huet72,Huet73]

Huet's Constrained Resolution $\mathcal{CR}$

Completeness Results

- [Huet72,Huet73] analyses completeness of $\mathcal{CR}$ formally only with respect to Andrews V-complexes, i.e. Huet verifies that the set of non-refutable sentences in $\mathcal{CR}$ is an abstract consistency class for V-complexes.
- Theorem (V-completeness of $\mathcal{CR}$): The calculus $\mathcal{CR}$ is complete with respect to the notion of V-complexes.

  Proof: [Huet72,Huet73]
- Theorem (Henkin completeness of $\mathcal{CR}$): The calculus $\mathcal{CR}$ is complete wrt. Henkin semantics provided that the infinitely many extensionality axioms are given.

  Proof: exercise

Example Proofs

Exercise: How are the following theorems proved in calculus $\mathcal{CR}$?

- Leibniz equality and η-equality:

$$\mathsf{f}_{\iota\rightarrow\iota} \doteq \lambda \mathsf{X}_\iota\text{.}\mathsf{f}\ \mathsf{X}$$

Example Proofs

Exercise: How are the following theorems proved in calculus $\mathcal{CR}$?

- Leibniz equality and η-equality:

$$\mathsf{f}_{\iota\to\iota} \doteq \lambda \mathsf{X}_\iota\text{.}\mathsf{f}\ \mathsf{X}$$

- The set of all red balls equals the set of all balls that are red: $\{\mathsf{X}|\mathsf{red}\ \mathsf{X} \wedge \mathsf{ball}\ \mathsf{X}\} = \{\mathsf{X}|\mathsf{ball}\ \mathsf{X} \wedge \mathsf{red}\ \mathsf{X}\}$. This problem can be encoded as

$$(\lambda \mathsf{X}_\iota\text{.}\mathsf{red}\ \mathsf{X} \wedge \mathsf{ball}\ \mathsf{X}) = (\lambda \mathsf{X}_\iota\text{.}\mathsf{ball}\ \mathsf{X} \wedge \mathsf{red}\ \mathsf{X})$$

Example Proofs

Exercise: How are the following theorems proved in calculus $\mathcal{CR}$?

- All unary logical operators $O_{o \to o}$ which map the propositions a and b to $\top$ consequently also map $a \wedge b$ to $\top$:

$$\forall O_{o \to o} \centerdot (O\ a_o) \wedge (O\ b_o) \Rightarrow (O\ (a_o \wedge b_o))$$

Example Proofs

Exercise: How are the following theorems proved in calculus $\mathcal{CR}$?

- In Henkin semantics the domain $\mathcal{D}_o$ of all Booleans contains exactly the truth values $\bot$ and $\top$. Consequently the domain of all mappings from Booleans to Booleans contains exactly contains in each Henkin model at most four elements. And because of the requirement, that the function domains in Henkin models must be rich enough such that every term has a denotation, it follows that $\mathcal{D}_{o\to o}$ contains exactly the pairwise distinct denotations of the following four terms: $\lambda X_o . X_o$, $\lambda X_o . \neg X_o$, $\lambda X_o . \bot$, and $\lambda X_o . \top$. This theorem can be formulated as follows (where $f_{o\to o}$ is a constant):

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

Approaches to Higher-Order Resolution: $\mathcal{ER}$

UNIVERSITÄT DES SAARLANDES

Extensional HO Resolution $\mathcal{ER}$

Clause normalization

$$\frac{C \lor [A \lor B]^T}{C \lor [A]^T \lor [B]^T}\ \lor^T \qquad \frac{C \lor [A \lor B]^F}{C \lor [A]^F}\ \lor^F_l \qquad \frac{C \lor [A \lor B]^F}{C \lor [B]^F}\ \lor^F_r$$

$$\frac{C \lor [\neg A]^T}{C \lor [A]^F}\ \neg^T \qquad \frac{C \lor [\neg A]^F}{C \lor [A]^T}\ \neg^F$$

$$\frac{C \lor [\Pi^\alpha A]^T \quad X_\alpha \text{ new variable}}{C \lor [A\ X]^T}\ \Pi^T$$

$$\frac{C \lor [\Pi^\alpha A]^F \quad sk_\alpha \text{ Skolem term}}{C \lor [A\ sk_\alpha]^F}\ \Pi^F$$

This rules may be combined into a single rule Cnf.

Extensional HO Resolution $\mathcal{ER}$

Resolution and Factorisation

$$\frac{[\mathsf{N}]^\alpha \vee \mathsf{C} \quad [\mathsf{M}]^\beta \vee \mathsf{D} \quad \alpha \neq \beta}{\mathsf{C} \vee \mathsf{D} \vee [\mathsf{N} \neq^? \mathsf{M}]}\ \mathsf{Res}$$

$$\frac{[\mathsf{N}]^\alpha \vee [\mathsf{M}]^\alpha \vee \mathsf{C} \quad \alpha \in \{\mathsf{T}, \mathsf{F}\}}{[\mathsf{N}]^\alpha \vee \mathsf{C} \vee [\mathsf{N} \neq^? \mathsf{M}]}\ \mathsf{Fac}$$

$$\frac{[\mathsf{Q}_\gamma \overline{\mathsf{U}^\mathsf{k}}]^\alpha \vee \mathsf{C} \quad \mathsf{P} \in \mathcal{GB}_\gamma^{\{\neg,\vee\}\cup\{\Pi^\beta|\beta\in\mathcal{T}^\mathsf{k}\}}}{[\mathsf{Q}_\gamma \overline{\mathsf{U}^\mathsf{k}}]^\alpha \vee \mathsf{C} \vee [\mathsf{Q} \neq^? \mathsf{P}]}\ \mathsf{Prim}^\mathsf{k}$$

Extensional HO Resolution $\mathcal{ER}$

(Pre-)unification rules

$$\frac{C \vee [M_{\alpha\to\beta} \neq^? N_{\alpha\to\beta}]^F \quad s_\alpha \text{ Skolem-Term}}{C \vee [M\ s \neq^? N\ s]}\ \text{Func}$$

$$\frac{C \vee [h\overline{U^n} \neq^? h\overline{V^n}]}{C \vee [U^1 \neq^? V^1] \vee \ldots \vee [U^n \neq^? V^n]}\ \text{Dec} \qquad \frac{C \vee [A \neq^? A]}{C}\ \text{Triv}$$

$$\frac{C \vee [F_\gamma\overline{U^n} \neq^? h\overline{V^n}] \quad G \in \mathcal{GB}^h_\gamma}{C \vee [F \neq^? G] \vee [F\overline{U^n} \neq^? h\overline{V^n}]}\ \text{Flex/Rigid}$$

$$\frac{C \vee E \quad E \text{ solved for } C}{Cnf(subst_E(C))}\ \text{Subst}$$

Extensional HO Resolution $\mathcal{ER}$

Extensionality rules

$$\frac{C \vee [M_o \neq^? N_o]^F}{Cnf(C \vee [M_o \Leftrightarrow N_o]^F)}\ \text{Equiv}$$

$$\frac{C \vee [M_\alpha \neq^? N_\alpha]^F \quad \alpha \in \{o, \iota\}}{Cnf(C \vee [\forall P_{\alpha\rightarrow o}.PM \Rightarrow PN]^F)}\ \text{Leib}$$

Extensional HO Resolution $\mathcal{ER}$

Extensionality Treatment

- Instead of adding infinitely many extensionality axioms to the search space $\mathcal{CR}$ provides two new extensionality rules which closely connect refutation search and eager unification.

Extensional HO Resolution $\mathcal{ER}$

Extensionality Treatment

- Instead of adding infinitely many extensionality axioms to the search space $\mathcal{CR}$ provides two new extensionality rules which closely connect refutation search and eager unification.
- The idea is to allow for recursive calls from higher-order unification to the overall refutation process.

Extensional HO Resolution $\mathcal{ER}$

Extensionality Treatment

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

Extensional HO Resolution $\mathcal{ER}$

Proof Search

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

Extensional HO Resolution $\mathcal{ER}$

Proof Search

- Initially the proof problem is negated and normalised. The main proof search then closely interleaves the refutation process on resolution layer and unification, i.e. the main proof search rules Res, Fac, and Prim and the unification rules are integrated at a common conceptual level. The calls from unification to the overall refutation process with rules *Leib* and *Equiv* introduce new clauses into the search space which can be resolved against already given ones.
- The following figure graphically illustrates the main ideas of the proof search in $\mathcal{ER}$.

interleaved proof search & unification

Ex.: Extensional HO Resolution $\mathcal{ER}$

$$\forall B_{\alpha\to o}, C_{\alpha\to o}, D_{\alpha\to o}. B \cup (C \cap D) = (B \cup C) \cap (B \cup D)$$

Negation and definition expansion with

$\cup = \lambda A_{\alpha\to o}, B_{\alpha\to o}, X_\alpha. (A\ X) \lor (B\ X) \qquad \cap = \lambda A_{\alpha\to o}, B_{\alpha\to o}, X_\alpha. (A\ X) \land (B\ X)$

leads to:

$$C_1 : [\lambda X_\alpha. (b\ X) \lor ((c\ X) \land (d\ X)) \neq^? \lambda X_\alpha. ((b\ X) \lor (c\ X)) \land ((b\ X) \lor (d\ X))]$$

Goal directed functional and Boolean extensionality treatment:

$$C_2 : [(b\ x) \lor ((c\ x) \land (d\ x)) \Leftrightarrow ((b\ x) \lor (c\ x)) \land ((b\ x) \lor (d\ x))]^F$$

Clause normalization results then in a pure propositional, i.e. decidable, set of clauses. Only these clauses are still in the search space of LEO(in total there are 33 clauses generated and LEO finds the proof on a 2,5GHz PC in 820ms).

Similar proof in case of embedded propositions:

$$\forall P_{(\alpha\to o)\to o}, B_{\alpha\to o}, C_{\alpha\to o}, D_{\alpha\to o}. P(B \cup (C \cap D)) \Rightarrow P((B \cup C) \cap (B \cup D))$$

Ex.: Extensional HO Resolution $\mathcal{ER}$

$$\forall P_{o \to o} . (P\ a_o) \land (P\ b_o) \Rightarrow (P\ (a_o \land b_o))$$

Negation and clause normalization

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

Resolution between $\mathcal{C}_1$ and $\mathcal{C}_3$ and between $\mathcal{C}_2$ and $\mathcal{C}_3$

$$\mathcal{C}_4 : [p\ a \neq^? p\ (a \land b)] \qquad \mathcal{C}_5 : [p\ b \neq^? p\ (a \land b)]$$

Decomposition

$$\mathcal{C}_6 : [a \neq^? (a \land b)] \qquad \mathcal{C}_7 : [b \neq^? (a \land b)]$$

Recursive call of proof process with rules Equiv and Cnf

$$\mathcal{C}_8 : [a]^F \lor [b]^F \qquad \mathcal{C}_9 : [a]^T \lor [b]^T \qquad \mathcal{C}_{10} : [a]^T \qquad \mathcal{C}_{11} : [b]^T$$

Ex.: Extensional HO Resolution $\mathcal{ER}$

Further small examples which test Henkin completeness:

$$\forall F_{o \to o} . (F \doteq \lambda X_o . X_o) \vee (F \doteq \lambda X_o . \neg X_o) \vee (F \doteq \lambda X_o . \bot) \vee (F \doteq \lambda X_o . \top)$$

$$\forall H_{o \to o} . H \bot \doteq H \, (H \top \doteq H \bot)$$

$$\ldots$$