Computer Supported Modeling and Reasoning

David Basin, Achim D. Brucker, Jan-Georg Smaus, and Burkhart Wolff

April 2005

http://www.infsec.ethz.ch/education/permanent/csmr/

Isabelle: Resolution

Burkhart Wolff

Higher-Order Unification

The typed λ-calculus is the syntactic device of Isabelle's built-in metalogic.

Higher-Order Unification

The typed λ-calculus is the syntactic device of Isabelle's built-in metalogic.The used version of the λ-calculus contains also "metavariables".

Higher-Order Unification

The typed λ-calculus is the syntactic device of Isabelle's built-in metalogic.The used version of the λ-calculus contains also "metavariables". Logically equivalent to free variables, they can be arbitrarily instantiated by terms or formulas.

Higher-Order Unification

The typed λ-calculus is the syntactic device of Isabelle's built-in metalogic.The used version of the λ-calculus contains also "metavariables". Logically equivalent to free variables, they can be arbitrarily instantiated by terms or formulas. They can be viewed as "delayed substitutions", that were provided during a proof at need.

Higher-Order Unification

The typed λ-calculus is the syntactic device of Isabelle's built-in metalogic.The used version of the λ-calculus contains also "metavariables". Logically equivalent to free variables, they can be arbitrarily instantiated by terms or formulas. They can be viewed as "delayed substitutions", that were provided during a proof at need.

This means: Isabelle´s proof engine will

Higher-Order Unification

The typed λ-calculus is the syntactic device of Isabelle's built-in metalogic.The used version of the λ-calculus contains also "metavariables". Logically equivalent to free variables, they can be arbitrarily instantiated by terms or formulas. They can be viewed as "delayed substitutions", that were provided during a proof at need.

This means: Isabelle´s proof engine will

- rename metavariables

Higher-Order Unification

The typed λ-calculus is the syntactic device of Isabelle's built-in metalogic.The used version of the λ-calculus contains also "metavariables". Logically equivalent to free variables, they can be arbitrarily instantiated by terms or formulas. They can be viewed as "delayed substitutions", that were provided during a proof at need.

This means: Isabelle´s proof engine will

- rename metavariables
- unify metavariables

during rule application.

What is Higher-Order Unification?

Unification of terms e, e':

find substitution θ for metavariables such that

$\theta(e) =_{\alpha\beta\eta} \theta(e')$.

Examples:

$$\begin{aligned} ?X + ?Y &=_{\alpha\beta\eta} x + x \\ ?P(x) &=_{\alpha\beta\eta} x + x \\ f(?X x) &=_{\alpha\beta\eta} ?Y\, x \end{aligned}$$

What is Higher-Order Unification?

Unification of terms e, e':
find substitution θ for metavariables such that
$\theta(e) =_{\alpha\beta\eta} \theta(e')$.

Examples:

$$\begin{aligned} ?X + ?Y &=_{\alpha\beta\eta} x + x \\ ?P(x) &=_{\alpha\beta\eta} x + x \\ f(?X x) &=_{\alpha\beta\eta} ?Y\, x \end{aligned}$$

Why higher-order? Metavariables may be instantiated to functions, e.g. $[?P \leftarrow \lambda y. y + y]$.

Higher-Order Unification: Some Facts

- Unification modulo $\alpha\beta\eta$ (HO-unification) is semi-decidable for the (monomorphic) typed λ-calculus.

Higher-Order Unification: Some Facts

- Unification modulo $\alpha\beta\eta$ (HO-unification) is semi-decidable for the (monomorphic) typed λ-calculus.
- Isabelle´s Unification modulo $\alpha\beta\eta$ is incomplete.

Higher-Order Unification: Some Facts

- Unification modulo $\alpha\beta\eta$ (HO-unification) is semi-decidable for the (monomorphic) typed λ-calculus.
- Isabelle´s Unification modulo $\alpha\beta\eta$ is incomplete.
- HO-unification is well-behaved for most practical cases.

Higher-Order Unification: Some Facts

- Unification modulo $\alpha\beta\eta$ (HO-unification) is semi-decidable for the (monomorphic) typed λ-calculus.
- Isabelle´s Unification modulo $\alpha\beta\eta$ is incomplete.
- HO-unification is well-behaved for most practical cases.
- Important fragments (like HO-patterns) are decidable.

Higher-Order Unification: Some Facts

- Unification modulo $\alpha\beta\eta$ (HO-unification) is semi-decidable for the (monomorphic) typed λ-calculus.
- Isabelle´s Unification modulo $\alpha\beta\eta$ is incomplete.
- HO-unification is well-behaved for most practical cases.
- Important fragments (like HO-patterns) are decidable.
- HO-unification has possibly infinitely many solutions.

Three Sections on Deduction Techniques: Outline

- Resolution

Three Sections on Deduction Techniques: Outline

- Resolution
- Proof search

Three Sections on Deduction Techniques: Outline

- Resolution
- Proof search
- Term rewriting

Three Sections on Deduction Techniques: Outline

- Resolution
- Proof search
- Term rewriting

Resolution

Resolution is the basic mechanism for transforming proof states in Isabelle in order to construct a proof.

It involves unifying a certain part of the current goal (state) with a certain part of a rule, and replacing that part of the current goal.

We now look at several variants of resolution.

Resolution (`rule`, as in Prolog)

$$\frac{\phi_1 \quad \cdots \quad \phi_i \quad \cdots \quad \phi_n}{\psi}$$

$\phi_1, \ldots, \phi_n$ are current subgoals and ψ is original goal. Isabelle displays

Level ... (n subgoals)
ψ
1. ϕ_1
⋮
n. ϕ_n

Resolution (`rule`, as in Prolog)

$$\frac{\alpha_1 \cdots \alpha_m}{\beta}$$

$$\frac{\phi_1 \quad \cdots \quad \phi_i \quad \cdots \quad \phi_n}{\psi}$$

$\phi_1, \ldots, \phi_n$ are current subgoals and ψ is original goal. Isabelle displays

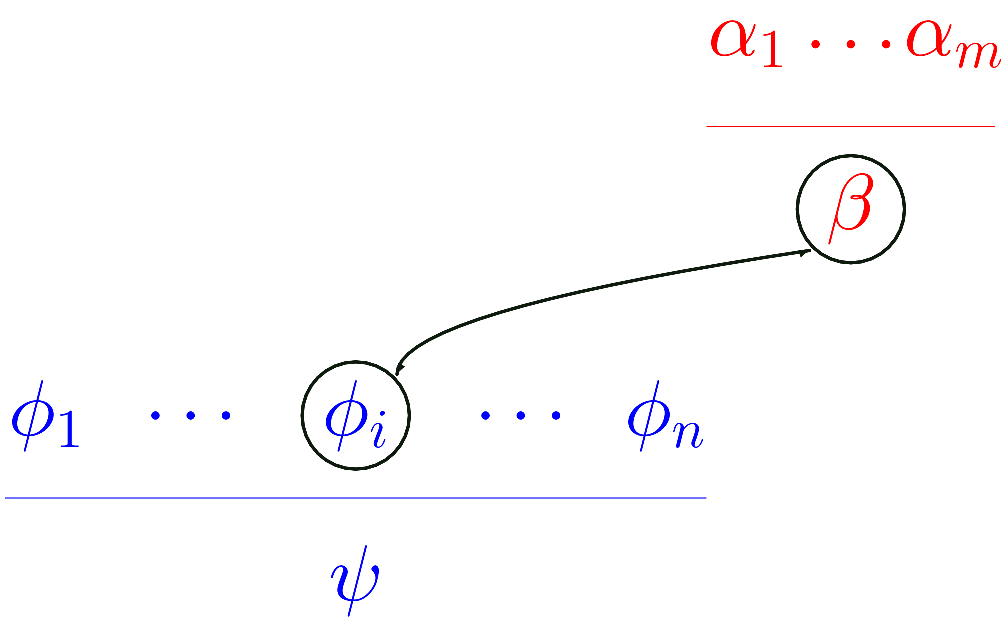
`Level ...` (n `subgoals`)
ψ
1. ϕ_1

⋮

n. ϕ_n

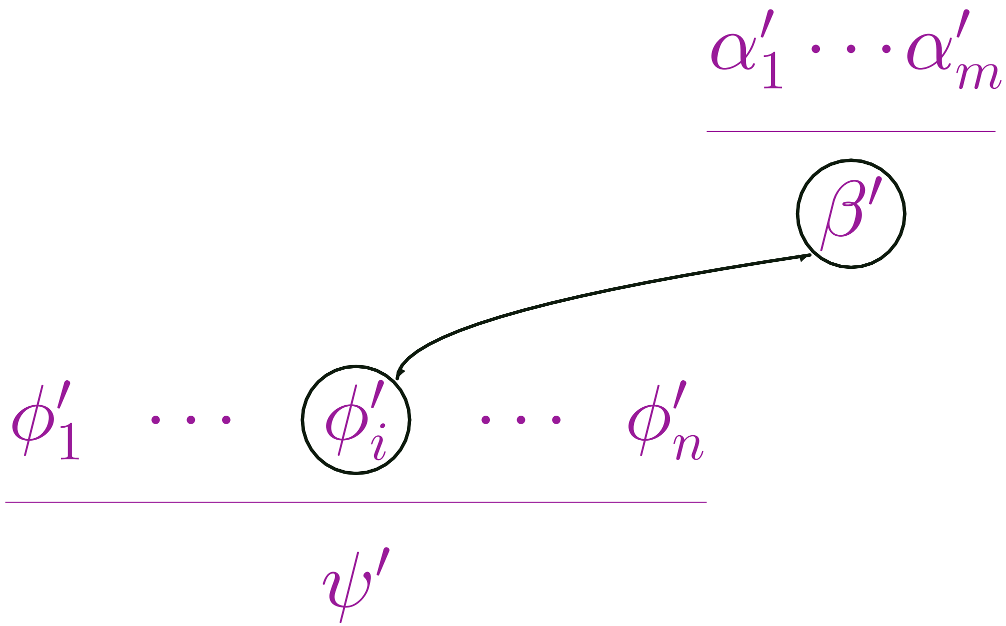
$[\![\alpha_1; \ldots; \alpha_m]\!] \Longrightarrow \beta$ is rule.

Resolution (`rule`, as in Prolog)



Simple scenario where ϕ_i has no premises. Now β must be unifiable with selected subgoal ϕ_i.

Resolution (`rule`, as in Prolog)



Simple scenario where ϕ_i has no premises. Now β must be unifiable with selected subgoal ϕ_i.

We apply the unifier ($'$)

Resolution (`rule`, as in Prolog)

$$\frac{\phi_1' \cdots \alpha_1' \cdots \alpha_m' \cdots \phi_n'}{\psi'}$$

Simple scenario where ϕ_i has no premises. Now β must be unifiable with selected subgoal ϕ_i.

We apply the unifier ($'$)

We replace ϕ_i' by the premises of the rule.

Resolution (with Lifting over Parameters)

$$\frac{\phi_1 \qquad \cdots \qquad \bigwedge x.\phi_i \qquad \cdots \qquad \phi_n}{\psi}$$

Now suppose the i'th (selected) subgoal is preceded by $\bigwedge$ (metalevel universal quantifier).

Resolution (with Lifting over Parameters)

$$\frac{\alpha_1 \quad \cdots \alpha_m}{\beta}$$

$$\frac{\phi_1 \quad \cdots \quad \bigwedge x.\phi_i \quad \cdots \quad \phi_n}{\psi}$$

Rule

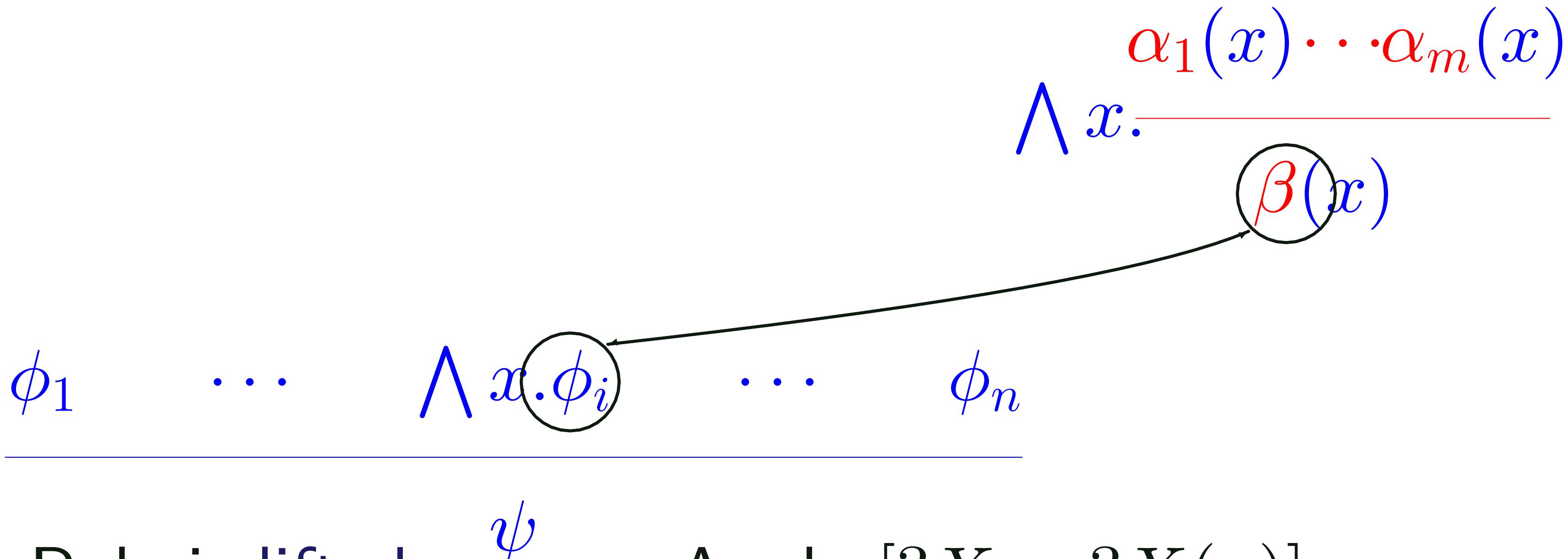
Resolution (with Lifting over Parameters)

$$\bigwedge x. \frac{\alpha_1(x) \cdots \alpha_m(x)}{\beta(x)}$$

$$\frac{\phi_1 \qquad \cdots \qquad \bigwedge x. \phi_i \qquad \cdots \qquad \phi_n}{\psi}$$

Rule is lifted over x: Apply $[?X \leftarrow ?X(x)]$.

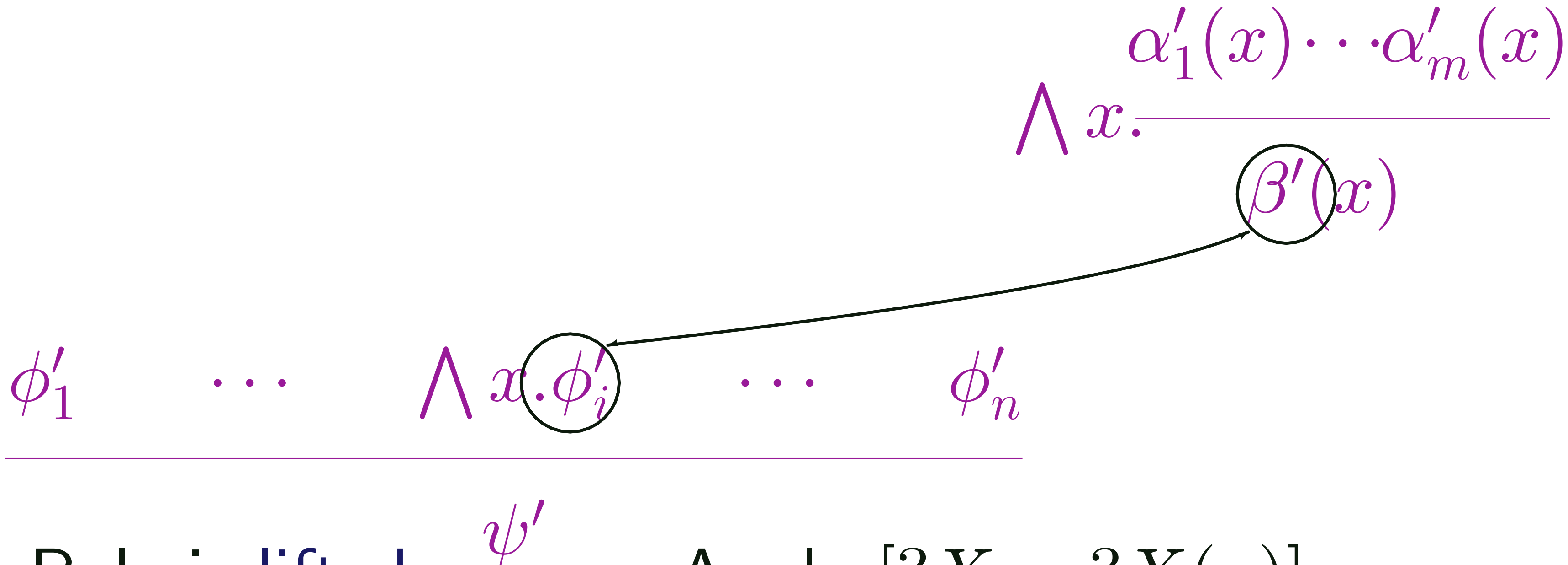
Resolution (with Lifting over Parameters)



Rule is lifted over x: Apply $[?X \leftarrow ?X(x)]$.

As before, β must be unifiable with ϕ_i;

Resolution (with Lifting over Parameters)



Rule is lifted over x: Apply $[?X \leftarrow ?X(x)]$.

As before, β must be unifiable with ϕ_i; apply the unifier.

Resolution (with Lifting over Parameters)

$$\frac{\phi'_1 \cdots \bigwedge x.\alpha'_1[x] \cdots \bigwedge x.\alpha'_m[x] \cdots \phi'_n}{\psi'}$$

Rule is lifted over x: Apply $[?X \leftarrow ?X(x)]$.

As before, β must be unifiable with ϕ_i; apply the unifier.

We replace ϕ'_i by the premises of the rule. $\alpha'_1, \ldots, \alpha'_m$ are preceded by $\bigwedge x$.

Resolution (with Lifting over Assumptions)

$$\frac{\phi_1 \quad \cdots \quad \begin{matrix}[\phi_{i1} \cdots \phi_{ik_i}] \\ \vdots \\ \phi_i\end{matrix} \quad \cdots \quad \phi_n}{\psi}$$

Now, suppose the i'th (selected) subgoal has assumptions $\phi_{i1}, \ldots, \phi_{ik_i}$.

Resolution (with Lifting over Assumptions)

$$\frac{\alpha_1 \quad \cdots \quad \alpha_m}{\beta}$$

$$\frac{\phi_1 \quad \cdots \quad \begin{array}{c}[\phi_{i1} \cdots \phi_{ik_i}] \\ \vdots \\ \phi_i\end{array} \quad \cdots \quad \phi_n}{\psi}$$

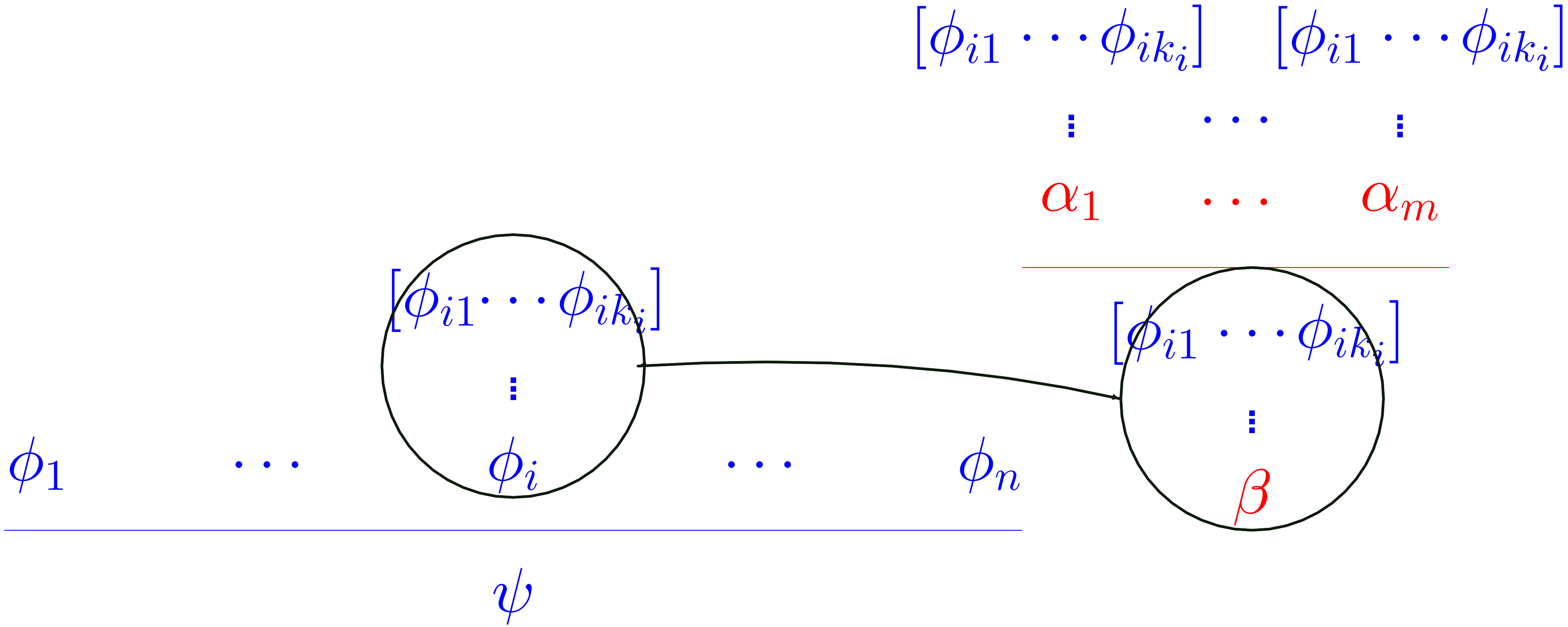
As before, we have a rule. In general β is not unifiable with the i'th subgoal, even assuming that β is unifiable with ϕ_i.

Resolution (with Lifting over Assumptions)

$$\dfrac{\phi_1 \quad \cdots \quad \begin{array}{c}[\phi_{i1} \cdots \phi_{ik_i}] \\ \vdots \\ \phi_i\end{array} \quad \cdots \quad \phi_n}{\psi} \qquad \begin{array}{c}\dfrac{\begin{array}{c}[\phi_{i1} \cdots \phi_{ik_i}] \\ \vdots \\ \alpha_1\end{array} \quad \begin{array}{c}\cdots \\ \\ \cdots\end{array} \quad \begin{array}{c}[\phi_{i1} \cdots \phi_{ik_i}] \\ \vdots \\ \alpha_m\end{array}}{\begin{array}{c}[\phi_{i1} \cdots \phi_{ik_i}] \\ \vdots \\ \beta\end{array}}\end{array}$$

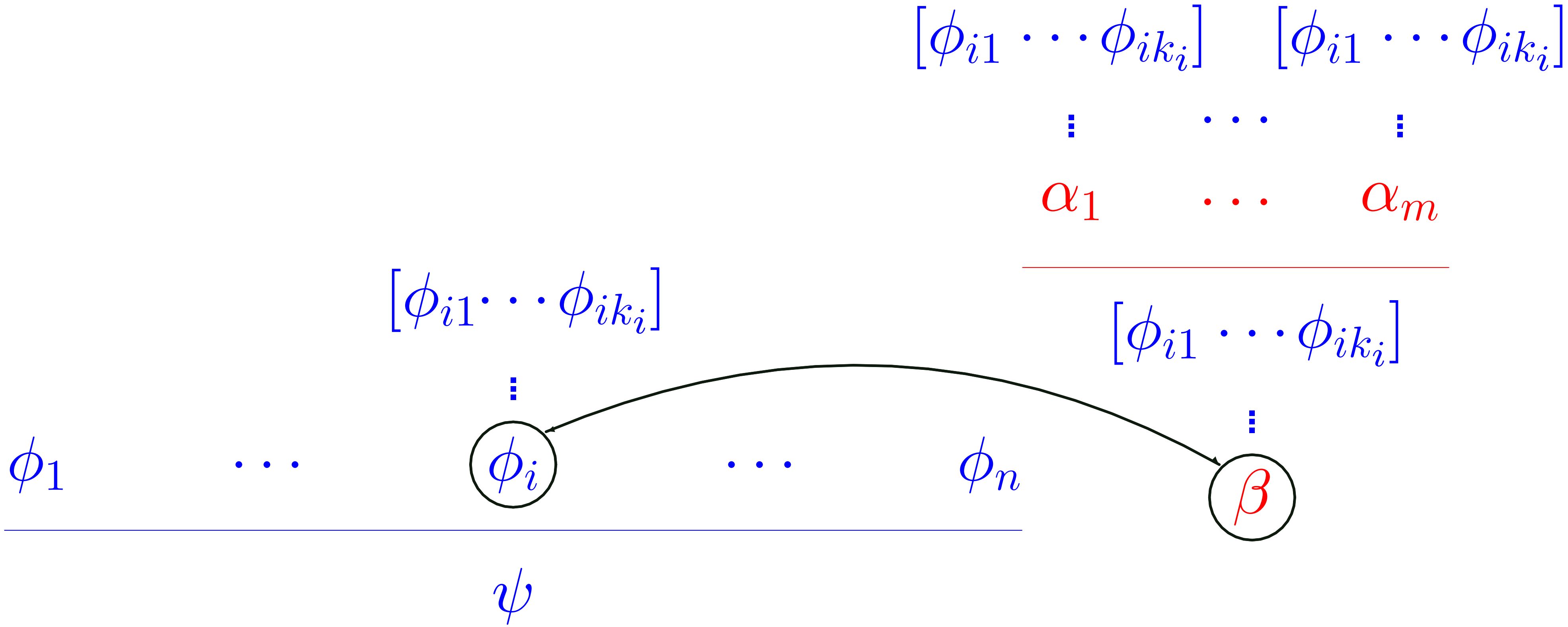
Rule must be lifted over assumptions. No unification so far!

Resolution (with Lifting over Assumptions)



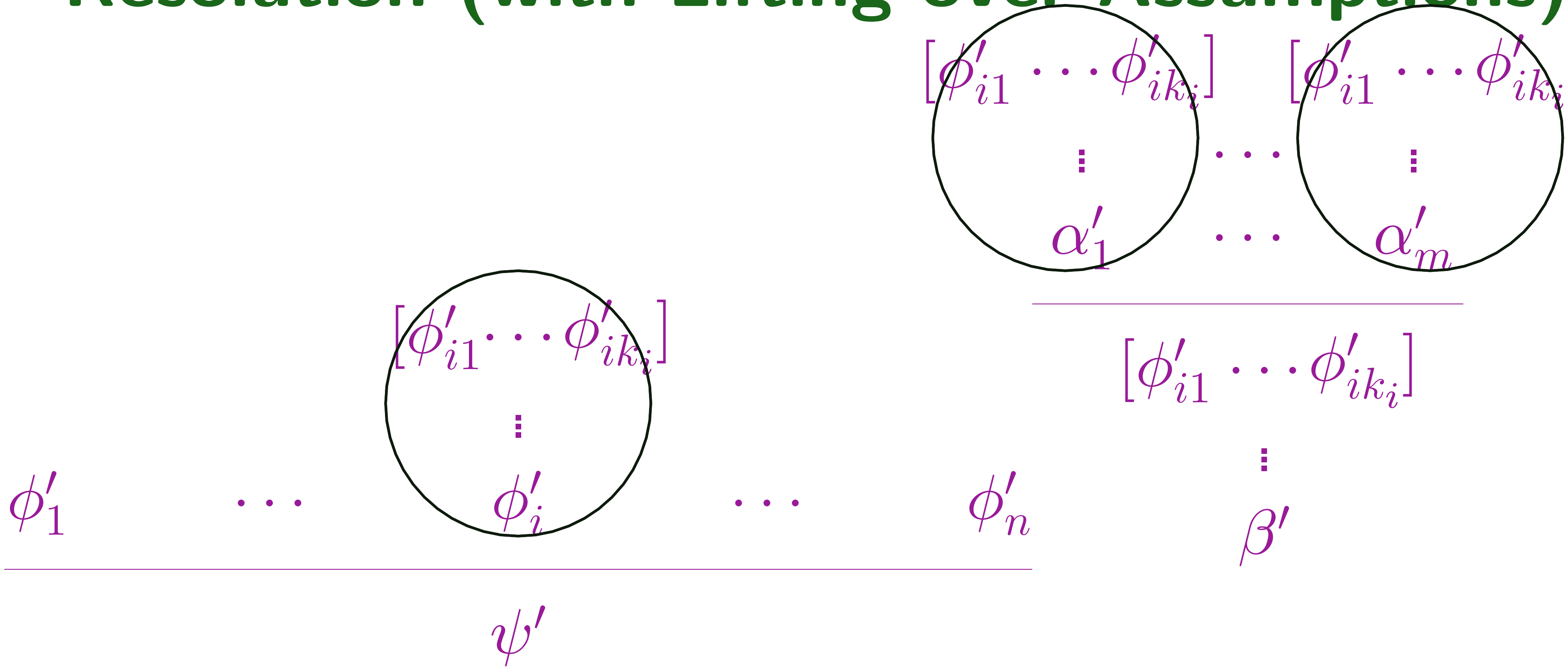
Now, subgoal and rule conclusion (below the bar) are unifiable.

Resolution (with Lifting over Assumptions)



Now, subgoal and rule conclusion (below the bar) are unifiable. Non-trivially, β must be unifiable with ϕ_i.

Resolution (with Lifting over Assumptions)



We apply the unifier.

Resolution (with Lifting over Assumptions)

$$\dfrac{\phi'_1 \cdots \phi'_{i-1} \quad \begin{matrix}[\phi'_{i1} \cdots \phi'_{ik_i}] \\ \vdots \\ \alpha'_1\end{matrix} \quad \begin{matrix} \\ \cdots \\ \cdots \end{matrix} \quad \begin{matrix}[\phi'_{i1} \cdots \phi'_{ik_i}] \\ \vdots \\ \alpha'_m\end{matrix} \quad \phi'_{i+1} \cdots \phi'_n}{\psi'}$$

We replace the subgoal.

Rule Premises Containing $\Longrightarrow$

$$\cfrac{\phi'_1 \cdots \qquad \cfrac{[\phi'_{i1} \cdots \phi'_{ik_i}]}{\vdots \atop \alpha'_j} \qquad \cdots \phi'_n}{\psi'}$$

What if some α_j has the form $[\![\gamma_1; \ldots; \gamma_l]\!] \Longrightarrow \delta$?

Rule Premises Containing $\Longrightarrow$

$$\cfrac{\phi'_1 \quad \cdots \quad \begin{matrix}[\phi'_{i1} \cdots \phi'_{ik_i}] \\ \vdots \\ [\![\gamma'_1; \ldots ; \gamma'_l]\!] \Longrightarrow \delta'\end{matrix} \quad \cdots \quad \phi'_n}{\psi'}$$

Is this what we get?

Rule Premises Containing $\Longrightarrow$

$$\cfrac{\phi'_1 \cdots \quad \begin{array}{c}[\phi'_{i1} \cdots \phi'_{ik_i} \; \gamma'_1 \cdots \gamma'_l] \\ \vdots \\ \delta'\end{array} \quad \cdots \phi'_n}{\psi'}$$

Is this what we get?

Well, we write $\vdots$ for $\Longrightarrow$, and use $A \Longrightarrow B \Longrightarrow C \equiv [\![A; B]\!] \Longrightarrow C$.

Elimination-Resolution

$$\frac{\alpha_1 \cdots \alpha_m}{\beta}$$

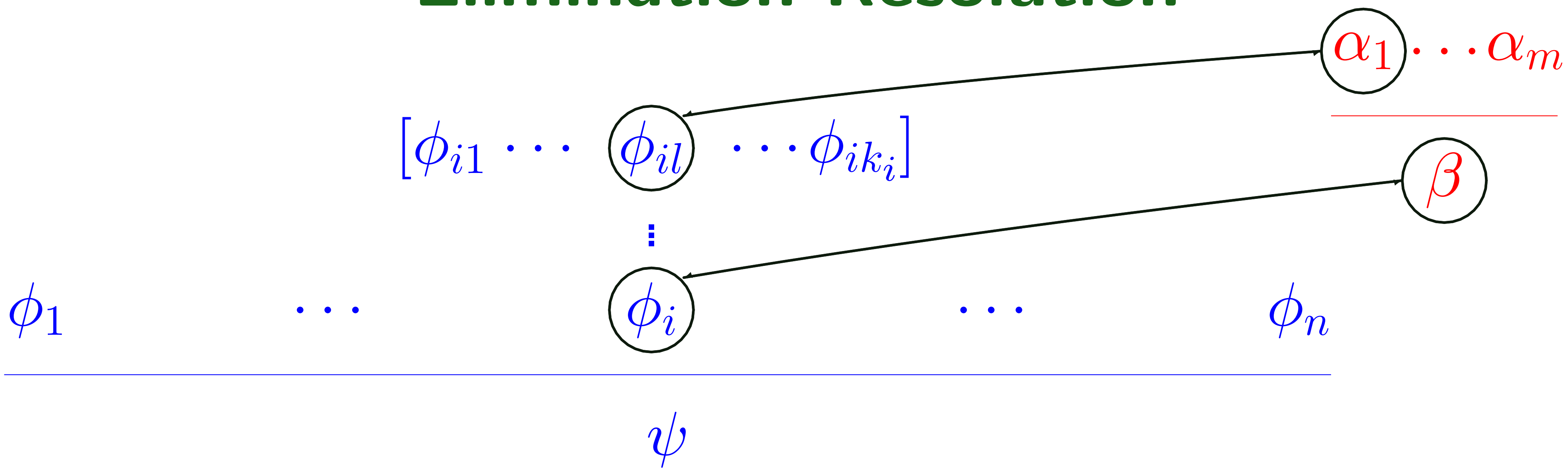
$$[\phi_{i1} \cdots \quad \phi_{il} \quad \cdots \phi_{ik_i}]$$

$$\vdots$$

$$\frac{\phi_1 \qquad \cdots \qquad \phi_i \qquad \cdots \qquad \phi_n}{\psi}$$

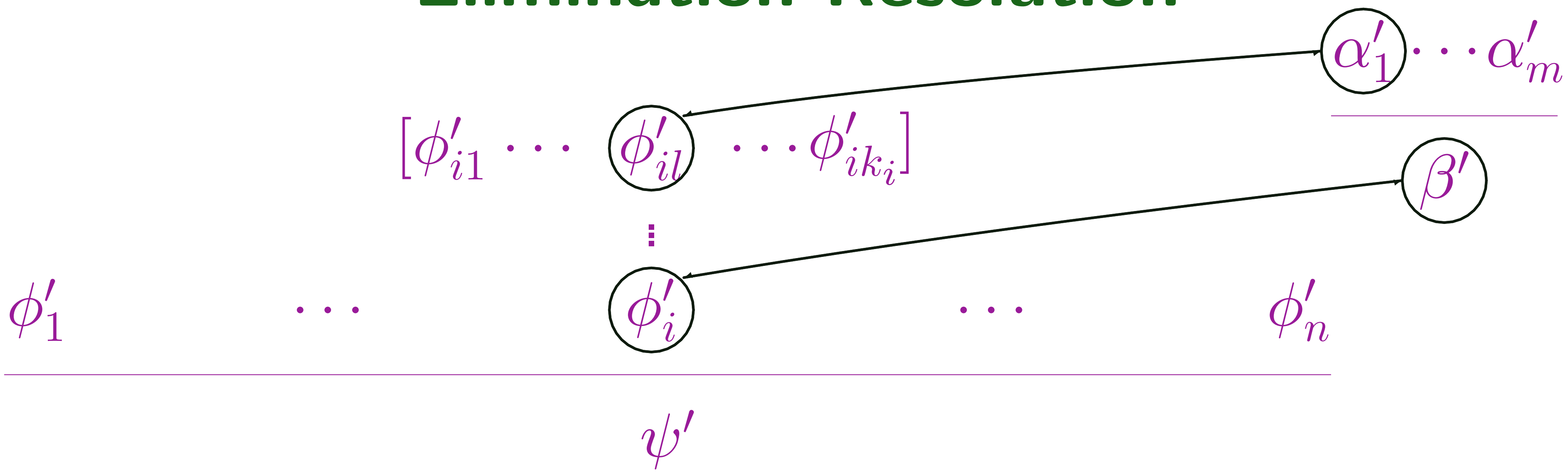
Same scenario as before

Elimination-Resolution



Same scenario as before, but now β must be unifiable with ϕ_i, and α_1 must be unifiable with ϕ_{il}, for some l.

Elimination-Resolution



Same scenario as before, but now β must be unifiable with ϕ_i, and α_1 must be unifiable with ϕ_{il}, for some l.

Apply the unifier.

Elimination-Resolution

$$\dfrac{\phi'_1 \quad \cdots \quad \phi'_{i-1} \quad \overset{\displaystyle [\phi'_{i1} \cdots \phi'_{i,l-1}, \phi'_{i,l+1} \cdots \phi'_{ik_i}]}{\overset{\vdots}{\alpha'_2}} \quad \cdots \quad \overset{\displaystyle [\phi'_{i1} \cdots \phi'_{i,l-1}, \phi'_{i,l+1} \cdots \phi'_{ik_i}]}{\overset{\vdots}{\alpha'_m}} \quad \phi'_{i+1} \cdots \phi'_n}{\psi'}$$

Same scenario as before, but now β must be unifiable with ϕ_i, and α_1 must be unifiable with ϕ_{il}, for some l.

Apply the unifier.

We replace ϕ'_i by the premises of the rule except first premise. $\alpha'_2, \ldots, \alpha'_m$ inherit the assumptions of ϕ'_i, except ϕ'_{il}.

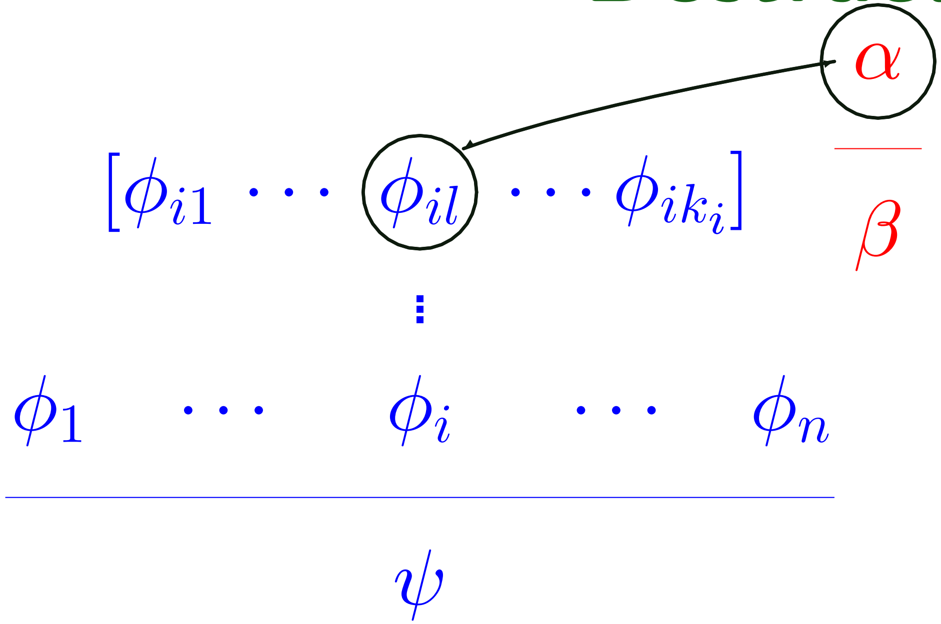
Destruct-Resolution

$$\frac{\alpha}{\beta}$$

$$\frac{\phi_1 \quad \cdots \quad \begin{matrix}[\phi_{i1} \cdots \phi_{il} \cdots \phi_{ik_i}] \\ \vdots \\ \phi_i\end{matrix} \quad \cdots \quad \phi_n}{\psi}$$

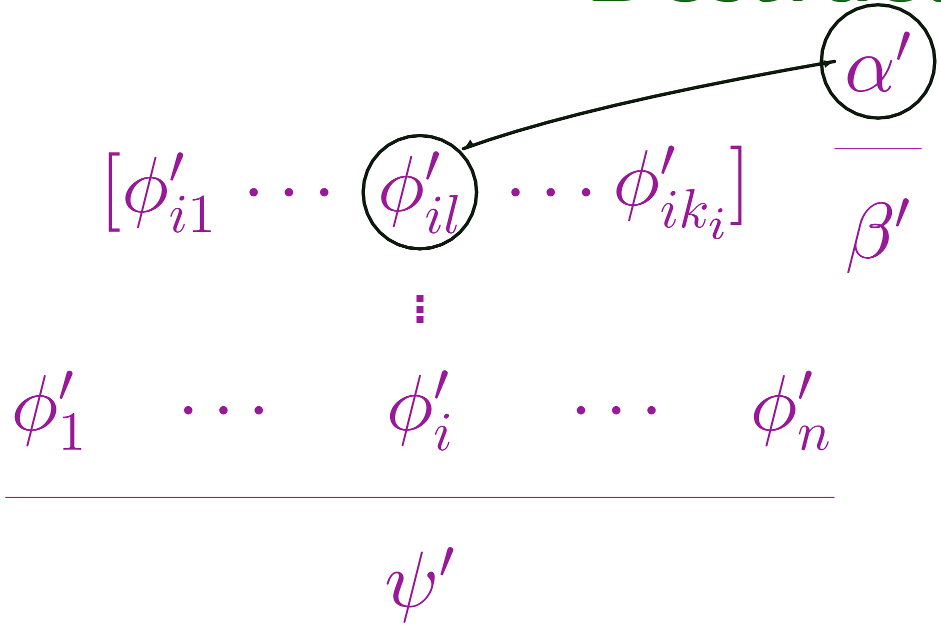
Simple rule

Destruct-Resolution



Simple rule, and α must be unifiable with ϕ_{il}, for some l.

Destruct-Resolution



Simple rule, and α must be unifiable with ϕ_{il}, for some l.

We apply the unifier.

Destruct-Resolution

$$\frac{\begin{array}{ccccc} & [\phi'_{i1} \cdots & \beta' & \cdots \phi'_{ik_i}] & \\ & & \vdots & & \\ \phi'_1 & \cdots & \phi'_i & \cdots & \phi'_n \end{array}}{\psi'}$$

Simple rule, and α must be unifiable with ϕ_{il}, for some l.

We apply the unifier.

We replace premise ϕ'_{il} with the conclusion of the rule.

Summary on Resolution

- Build proof resembling sequent style notation;
- technically: replace goals with rule premises, or goal premises with rule conclusions;
- metavariables and unification to obtain appropriate instance of rule, delay commitments;
- lifting over parameters and assumptions;
- various techniques to manipulate premises or conclusions, as convenient: `rule`, `erule`, `drule`.

More Detailed Explanations

Prolog

Prolog is a logic programming language.

The computation mechanism of Prolog is resolution of a current goal (corresponding to our $\phi_1, \ldots, \phi_n$) with a Horn clause (corresponding to our $[\![\alpha_1; \ldots; \alpha_m]\!] \Longrightarrow \beta$). It is possible to write a little tactic program in Isabelle that "implements" a (Higher-order) Prolog interpreter.

Simple ϕ_i

ϕ_i is the selected subgoal. Isabelle kernel tactics can address with the i directly a selected subgoal. In the ISAR language, one writes:

$$\texttt{prefer i}; \texttt{apply } (\textit{tactic rule})$$

With `defer i` a subgoal may be pushed towards the end of the subgoal list.

We assume here that ϕ_i is a formula, i.e., it contains no $\Longrightarrow$ (metalevel implication). The form of the other subgoals $\phi_1, \ldots, \phi_{i-1}, \phi_{i+1}, \ldots, \phi_n$ is arbitrary.

Prime ($'$)

In all illustrations that follow, we use $'$ to suggest the application of the appropriate unifier.

Metalevel Universal Quantification

$\bigwedge$ is the metalevel universal quantification (also written !!). If a goal is preceded by $\bigwedge x$, this means that Isabelle treats x as a fresh free variable (also in user defined substitutions).

Lifting over Parameters

The metavariables of the rule are made dependent on x. That is to say, each metavariable $?X$ is replaced by a $?X(x)$. You may also say that $?X$ is now a Skolem function of x.

This process is called lifting the rule over the parameter x.

Lifting over Assumptions

Each premise of the rule, as well as the conclusion of the rule, are preceded by the assumptions $[\![\phi_{i1}, \ldots, \phi_{ik_i}]\!]$ of the current subgoals. Actually, the rule

$$\frac{\begin{array}{ccc} [\phi_{i1} \cdots \phi_{ik_i}] & & [\phi_{i1} \cdots \phi_{ik_i}] \\ \vdots & \cdots & \vdots \\ \alpha_1 & \cdots & \alpha_m \end{array}}{\begin{array}{c} [\phi_{i1} \cdots \phi_{ik_i}] \\ \vdots \\ \beta \end{array}}$$

may look different from any rules you have seen so far, but it can be formally derived from the rule:

$$\frac{\alpha_1 \quad \cdots \quad \alpha_m}{\beta}$$

The derived rule should be read as: If for all $j \in \{1, \ldots, m\}$, we can derive α_j from $\phi_{i1}, \ldots, \phi_{ik_i}$, then we can derive β from $\phi_{i1}, \ldots, \phi_{ik_i}$.

Unifiability

Still assuming that ϕ_i and β are unifiable.

A Trivial Unification

Both the subgoal and the conclusion of the lifted rule are preceded by assumptions $\phi_{i1}, \ldots, \phi_{ik_i}$. Hence the assumption list of the subgoal and the assumption list of the rule are trivially unifiable since they are identical.

Folding Assumptions

Generally, Isabelle makes no distinction between

$$[\![\psi_1; \ldots; \psi_n]\!] \Longrightarrow [\![\mu_1; \ldots; \mu_k]\!] \Longrightarrow \phi$$

and

$$[\![\psi_1; \ldots; \psi_n; \mu_1; \ldots; \mu_k]\!] \Longrightarrow \phi$$

and displays the second form. Semantically, this corresponds to the equivalence of $A_1 \wedge \ldots \wedge A_n \to B$ and $A_1 \to \ldots \to A_n \to B$.

We have seen this in the exercises.

Same as Resolution

So the scenario looks as for resolution with lifting over assumptions. However, this time we do not show the lifting over assumptions in our animation.

The Rationale of Elimination-Resolution

Elimination-resolution is used to transform a formula in the assumption list.

For example, if the current goal is

$$\dfrac{\begin{array}{c}[A \wedge B] \\ \vdots \\ B\end{array}}{A \wedge B \to B}$$

and the rule is

$$\dfrac{P \wedge Q \qquad \begin{array}{c}[P; Q] \\ \vdots \\ R\end{array}}{R}\ \wedge\text{-}E$$

then the result of elimination resolution is

$$\dfrac{\begin{matrix}[A; B] \\ \vdots \\ B\end{matrix}}{A \wedge B \rightarrow B}$$

Elimination resolution plays a key-role in case-distinction proofs and brings a forward proof element into backward proofs. The name of elimination resolution is motivated by the name for a particular type of rules in natural deduction calculi called elimination rules. Note that the first premise of a rule plays a distinguished role in elimination resolution.

The Rationale of Destruct-Resolution

Destruct-resolution is used to replace a formula in the assumption list by the conclusion of a rule.

For example, if the current goal is

$$\frac{\begin{array}{c}[A \wedge B]\\ \vdots \\ B\end{array}}{A \wedge B \to B}$$

and the rule is

$$\frac{P \wedge Q}{Q}\ \texttt{conjunct2}$$

then the result of destruct-resolution is

$$\dfrac{\begin{matrix}[B] \\ \vdots \\ B\end{matrix}}{A \wedge B \rightarrow B}$$

The name of destruction resolution is motivated by the name for a particular type of rules in natural deduction calculi called destruction rules.